REPORT DOCUMENTATION PAGE

AFRL-SR-BL-TR-98-

0774

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3.
	November, 1994	Final

4. TITLE AND SUBTITLE
USAF Summer Research Program - 1993 Summer Research Extension Program Final Reports, Volume 3, Rome Laboratory

5. FUNDING NUMBERS

6. AUTHORS
Gary Moore

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Research and Development Labs, Culver City, CA

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NI
4040 Fairfax Dr, Suite 500
Arlington, VA 22203-1613

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Contract Number: F4962-90-C-0076

12a. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The purpose of this program is to develop the basis for continuing research of interest to the Air Force at the institution of the faculty member; to stimulate continuing relations among faculty members and professional peers in the Air Force to enhance the research interests and capabilities of scientific and engineering educators; and to provide follow-on funding for research of particular promise that was started at an Air Force laboratory under the Summer Faculty Research Program. Each participant provided a report of their research, and these reports are consolidated into this annual report.

14. SUBJECT TERMS
AIR FORCE RESEARCH, AIR FORCE, ENGINEERING, LABORATORIES, REPORTS, UNIVERSITIES

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

UNITED STATES AIR FORCE

SUMMER RESEARCH PROGRAM -- 1993

SUMMER RESEARCH EXTENSION PROGRAM FINAL REPORTS

VOLUME 3

ROME LABORATORY

RESEARCH & DEVELOPMENT LABORATORIES

5800 Uplander Way

Culver City, CA 90230-6608

Program Director, RDL
Gary Moore

Program Manager, AFOSR
Major David Hart

Program Manager, RDL
Scott Licoscos

Program Administrator, RDL
Gwendolyn Smith

Program Administrator, RDL
Johnetta Thompson

Submitted to:

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH

Bolling Air Force Base

Washington, D.C.

November 1994

PREFACE

This volume is part of a five-volume set that summarizes the research of participants in the 1993 AFOSR Summer Research Extension Program (SREP). The current volume, Volume 3 of 5, presents the final reports of SREP participants at Rome Laborarory.

Reports presented in this volume are arranged alphabetically by author and are numbered consecutively -- e.g., 1-1, 1-2, 1-3; 2-1, 2-2, 2-3, with each series of reports preceded by a 35 page management summary. Reports in the five-volume set are organized as follows:

VOLUME	TITLE
1A	Armstrong Laboratory (part one)
1B	Armstrong Laboratory (part two)
2	Phillips Laboratory
3	Rome Laboratory
4A	Wright Laboratory (part one)
4B	Wright Laboratory (part two)
5	Arnold Engineering Development Center Frank J. Seiler Research Laboratory Wilford Hall Medical Center

1993 SREP FINAL REPORTS

Armstrong Laboratory

VOLUME 1A

1993 SREP FINAL REPORTS

Armstrong Laboratory

VOLUME 1B

1993 SREP FINAL REPORTS

Phillips Laboratory

VOLUME 2

1993 SREP FINAL REPORTS

Phillips Laboratory

VOLUME 2
cont'd

Report #	Report Title Author's University	Report Author
15	Particle Stimulation of Plasmas	Dr. Richard Murphy Physics
	University of Missouri, Kansas City, MO	PL/WS
16	A Universal Equation of State for Shock in Homogeneous Materials	Dr. Jon Shively Engineering & Computer
	California State University, Northridge, CA	PL/VT Science
17	Speed-Up of the Phase Diversity Method Via Reduced Region & Optimization Dimen.	Dr. Johanna Stenzel Arts & Sciences
	University of Houston, Victoria, TX	PL/LI
18	Analysis of Solwind P-78 Fragmentation Using Empirical And Analytical Codes	Dr. Arjun Tan Physics
	Alabama A & M University, Normal, AL	PL/WS
19	Experimental Investigations of Homogeneous and Heterogeneous Nucleation/Condensa	Dr. Philip Whitefield Physics
	University of Missouri, Rolla, MO	PL/LI

1993 SREP FINAL REPORTS

Rome Laboratory

VOLUME 3

1993 SREP FINAL REPORTS

Wright Laboratory

VOLUME 4A

Report #	Report Title Author's University	Report Author
1	Integrated Estimator/Guidance/Autopilot for Homing Missiles University of Missouri, Rolla, MO	Dr. S. Balakrishan Mechanical & Aerospace WL/MN Engineering
2	Studies of NTO Decomposition Memphis State University, Memphis, TN	Dr. Theodore Burkey Chemistry WL/MN
3	Investigation of Ray-Beam Basis Functions for Use with the Generalized Ray Expan Ohio State University, Columbus, OH	Dr. Robert Burkholder Electrical Engineering WL/AA
4	Wave Mechanics Modeling of Terminal Ballistics Phenomenology Louisiana Tech University, Ruston, LA	Dr. Eugene Callens, Jr. Mechanical and Industrial WL/MN Engineer
5	Modeling for Aeroelastic Parameter Estimation of Flexing Slender Bodies in a Bal University of California, Berkeley, CA	Dr. Gary Chapman Mechnical Engineering WL/MN
6	Using VHDL in VSL Bist Design Synthesis and its Application to 3-D Pixel Graphic Wright State University, Dayton, OH	Dr. Chien-In Chen Electrical Engineering WL/EL
7	Study of Part Quality and Shrinkage for Injection Molded Aircraft Transparencies Florida International University, Miami, FL	Dr. Joe Chow Industrial and Systems WL/FI Engineering
8	Implementation of Noise-Reducing Multiple-Source Schlieren Systems Purdue University, West Lafayette, IN	Dr. Steven Collicott Aeronautics and WL/FI Astronautical Engineering
9	Performing Target Classification Using Fussy Morphology Neural Networks Iowa State University, Ames, IA	Dr. Jennifer Davidson Electrical Engineering WL/MN
10	Turbulent Heat Transfer In Counter-Rotating Disk System University of Dayton, Dayton, OH	Dr. Jamie Ervin Mechanical and Aerospace WL/ML Engineering
11	Modelling of Biomaterials for Non-Linear Optical Applications University of Virginia, Charlottesville, VA	Dr. Barry Farmer Materials Science and WL/ML Engineering
12	Passive Ranging, Roll-angle Approximation, and Target Recognition for Fuze Appli Florida State University, Tallahassee, FL	Dr. Simon Foo Electrical Engineering WL/MN
13	A Role of Oxygen and Sulfur Compounds in Jet Fuel Deposit Formation Eastern Kentucky University, Richmond, KY	Ms. Ann Gillman Chemistry WL/PO
14	Effect of Aeroelasticity on Experimental Nonlinear Indicial Responses Measured Ohio University, Athens, OH	Dr. Gary Graham Mechanical Engineering WL/FI

1993 SREP FINAL REPORTS

Wright Laboratory

VOLUME 4B

1993 SREP FINAL REPORTS

Wright Laboratory

VOLUME 4B
cont'd

Report #	Report Title Author's University	Report Author
33	Detection of Internal Defects in Multilayered Plates By Lamb Wave Acoustic Micro Universtiy of Arizona, Tucson, AZ	Dr. Kundu Tribikram Civil Engineering and WL/ML Engineering
34	Wavelet Analysis of Ultrasonic Signals for Non-Destructive Evaluation of Composi University of Dayton, Dayton, OH	Dr. Theresa Tuthill Electrical Engineering WL/ML
35	Stochastic Modeling of MBE Growth of Compoud Semiconductors University of Nevada, Las Vegas, NV	Dr. Ramasubrama Venkatasubraman Electrical and Computer WL/ML Engineering
36	Performance Evaluation And Improvement of a Resonant DC Link Inverter With A Lim North Dakota State University, Fargo, ND	Dr. Subbaraya Yuvarajan Electrical Engineering WL/PO
37	Three Component LDV Measurements in a Swirl Combustor North Carolina State University, Raleigh, NC	Dr. Richard Gould Mechanical and Aerospace WL/PO Engineering

1993 SREP FINAL REPORTS

VOLUME 5

1993 SUMMER RESEARCH EXTENSION PROGRAM (SREP) MANAGEMENT REPORT

1.0 BACKGROUND

Under the provisions of Air Force Office of Scientific Research (AFOSR) contract F49620-90-C-0076, September 1990, Research & Development Laboratories (RDL), an 8(a) contractor in Culver City, CA, manages AFOSR's Summer Research Program. This report is issued in partial fulfillment of that contract (CLIN 0003AC).

The Summer Research Extension Program (SREP) is one of four programs AFOSR manages under the Summer Research Program. The Summer Faculty Research Program (SFRP) and the Graduate Student Research Program (GSRP) place college-level research associates in Air Force research laboratories around the United States for 8 to 12 weeks of research with Air Force scientists. The High School Apprenticeship Program (HSAP) is the fourth element of the Summer Research Program, allowing promising mathematics and science students to spend two months of their summer vacations working at Air Force laboratories within commuting distance from their homes.

SFRP associates and exceptional GSRP associates are encouraged, at the end of their summer tours, to write proposals to extend their summer research during the following calendar year at their home institutions. AFOSR provides funds adequate to pay for 75 SREP subcontracts. In addition, AFOSR has traditionally provided further funding, when available, to pay for additional SREP proposals, including those submitted by associates from Historically Black Colleges and Universities (HBCUs) and Minority Institutions (MIs). Finally, laboratories may transfer internal funds to AFOSR to fund additional SREPs. Ultimately the laboratories inform RDL of their SREP choices, RDL gets AFOSR approval, and RDL forwards a subcontract to the institution where the SREP associate is employed. The subcontract (see Appendix 1 for a sample) cites the SREP associate as the principal investigator and requires submission of a report at the end of the subcontract period.

Institutions are encouraged to share costs of the SREP research, and many do so. The most common cost-sharing arrangement is reduction in the overhead, fringes, or administrative charges institutions would normally add on to the principal investigator's or research associate's labor. Some institutions also provide other support (e.g., computer run time, administrative assistance, facilities and equipment or research assistants) at reduced or no cost.

When RDL receives the signed subcontract, we fund the effort initially by providing 90% of the subcontract amount to the institution (normally $18,000 for a $20,000 SREP). When we receive the end-of-research report, we evaluate it administratively and send a copy to the laboratory for a technical evaluation. When the laboratory notifies us the SREP report is acceptable, we release the remaining funds to the institution.

2.0 THE 1993 SREP PROGRAM

SELECTION DATA: A total of 719 faculty members (SFRP Associates) and 286 graduate students (GSRP associates) applied to participate in the 1992 Summer Research Program. From these applicants 185 SFRPs and 121 GSRPs were selected. The education level of those selected was as follows:

1992 SRP Associates, by Degree			
SFRP		GSRP	
PHD	MS	MS	BS
179	6	52	69

Of the participants in the 1992 Summer Research Program 90 percent of SFRPs and 25 percent of GSRPs submitted proposals for the SREP. Ninety proposals from SFRPs and ten from GSRPs were selected for funding, which equates to a selection rate of 54% of the SFRP proposals and of 34% for GSRP proposals.

1993 SREP: Proposals Submitted vs. Proposals Selected			
	Summer 1992 Participants	Submitted SREP Proposals	SREPs Funded
SFRP	185	167	90
GSRP	121	29	10
TOTAL	306	196	100

The funding was provided as follows:

Contractual slots funded by AFOSR	75
Laboratory funded	14
Additional funding from AFOSR	11
Total	100

Six HBCU/MI associates from the 1992 summer program submitted SREP proposals; six were selected (none were lab-funded; all were funded by additional AFOSR funds).

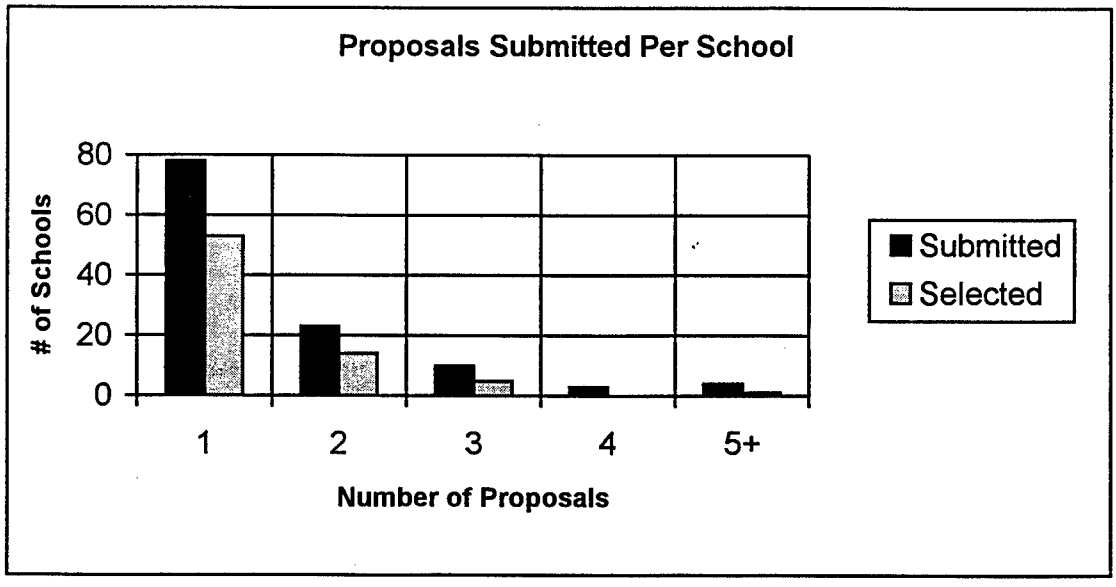
Proposals Submitted and Selected, by Laboratory		
	Applied	Selected
Air Force Civil Engineering Laboratory	9	4
Armstrong Laboratory	41	19
Arnold Engineering Development Center	12	4
Frank J. Seiler Research Laboratory	6	3
Phillips Laboratory	33	19
Rome Laboratory	31	13
Wilford Hall Medical Center	2	1
Wright Laboratory	62	37
TOTAL	196	100

Note: Phillips Laboratory funded 3 SREPs; Wright Laboratory funded 11; and AFOSR funded 11 beyond its contractual 75.

The 306 1992 Summer Research Program participants represented 135 institutions.

Institutions Represented on the 1992 SRP and 1993 SREP		
Number of schools represented in the Summer 92 Program	Number of schools represented in submitted proposals	Number of schools represented in Funded Proposals
135	118	73

Forty schools had more than one participant submitting proposals.



The selection rate for the 78 schools submitting 1 proposal (68%) was better than those submitting 2 proposals (61%), 3 proposals (50%), 4 proposals (0%) or 5+ proposals (25%). The 4 schools that submitted 5+ proposals accounted for 30 (15%) of the 196 proposals submitted.

Of the 196 proposals submitted, 159 offered institution cost sharing. Of the funded proposals which offered cost sharing, the minimum cost share was $1000.00, the maximum was $68,000.00 with an average cost share of $12,016.00.

Proposals and Institution Cost Sharing		
	Proposals Submitted	Proposals Funded
With cost sharing	159	82
Without cost sharing	37	18
Total	196	100

The SREP participants were residents of 41 different states. Number of states represented at each laboratory were:

States Represented, by Proposals Submitted/Selected per Laboratory		
	Proposals Submitted	Proposals Funded
Air Force Civil Engineering Laboratory	8	4
Armstrong Laboratory	21	13
Arnold Engineering Development Center	5	2
Frank J. Seiler Research Laboratory	5	3
Phillips Laboratory	16	14
Rome Laboratory	14	7
Wilford Hall Medical Center	2	1
Wright Laboratory	24	20

Eleven of the 1993 SREP Principal Investigators also participated in the 1992 SREP.

ADMINISTRATIVE EVALUATION: The administrative quality of the SREP associates' final reports was satisfactory. Most complied with the formatting and other instructions provided to them by RDL. Ninety seven final reports and two interim reports have been received and are included in this report. The subcontracts were funded by $1,991,623.00 of Air Force money. Institution cost sharing totaled $985,353.00.

TECHNICAL EVALUATION: The form used for the technical evaluation is provided as Appendix 2. ninety-two evaluation reports were received. Participants by laboratory versus evaluations submitted is shown below:

	Participants	Evaluations	Percent
Air Force Civil Engineering Laboratory	*	*	*
Armstrong Laboratory	23[1]	20	95.2
Arnold Engineering Development Center	4	4	100
Frank J. Seiler Research Laboratory	3	3	100
Phillips Laboratory	19[2]	18	100
Rome Laboratory	13	13	100
Wilford Hall Medical Center	1	1	100
Wright Laboratory	37	34	91.9
Total	100[3]	93	95.9

*AFCEL was combined with Wright Laboratory's Flight Dynamics Directorate and Armstrong Laboratories Environics Directorate in 1993. All four of AFCEL's SREP awards went to Armstrong Laboratories Environics Directorate, and their reports are included with Armstrong Lab.

Notes:

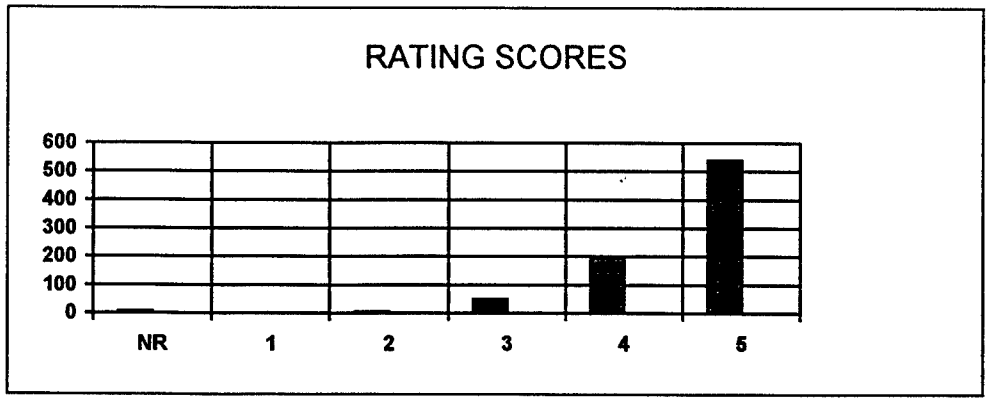
1: Research on two of the final reports was incomplete as of press time so there aren't any technical evaluations on them to process, yet. Percent complete is based upon 20/21=95.2%

2: One technical evaluation was not completed because one of the final reports was incomplete as of press time. Percent complete is based upon 18/18=100%

3: See notes 1 and 2 above. Percent complete is based upon 93/97=95.9%

The number of evaluations submitted for the 1993 SREP (95.9%) shows a marked improvement over the 1992 SREP submittals (65%).

PROGRAM EVALUATION: Each laboratory focal point evaluated ten areas (see Appendix 2) with a rating from one (lowest) to five (highest). The distribution of ratings was as follows:

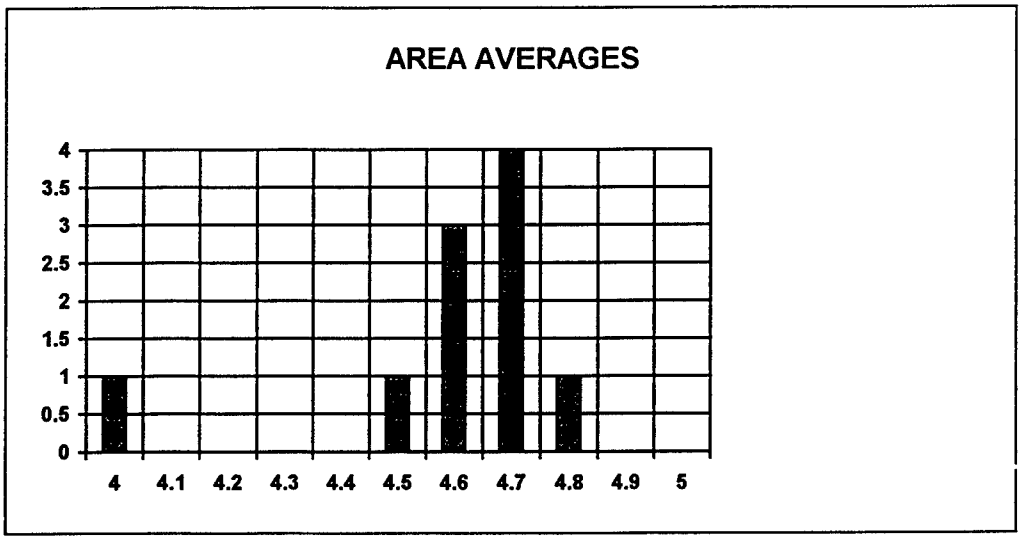


Rating	Not Rated	1	2	3	4	5
# Responses	7	1	7	62 (6%)	226 (25%)	617 (67%)

The 8 low ratings (one 1 and seven 2's) were for question 5 (one 2) "The USAF should continue to pursue the research in this SREP report" and question 10 (one 1 and six 2's) "The one-year period for complete SREP research is about right", in addition over 30% of the threes (20 of 62) were for question ten. The average rating by question was:

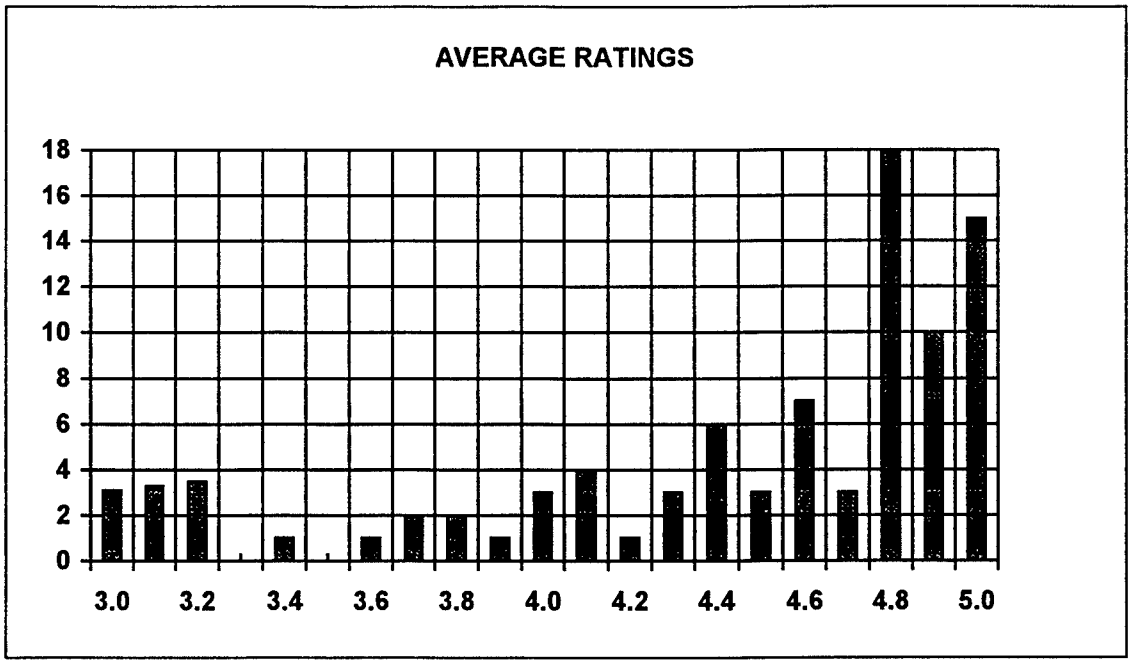
Question	1	2	3	4	5	6	7	8	9	10
Average	4.6	4.6	4.7	4.7	4.6	4.7	4.8	4.5	4.6	4.0

The distribution of the averages was:



Area 10 "the one-year period for complete SREP research is about right" had the lowest average rating (4.1). The overall average across all factors was 4.6 with a small sample standard deviation of 0.2. The average rating for area 10 (4.1) is approximately three sigma lower than the overall average (4.6) indicating that a significant number of the evaluators feel that a period of other than one year should be available for complete SREP research.

The average ratings ranged from 3.4 to 5.0. The overall average for those reports that were evaluated was 4.6. Since the distribution of the ratings is not a normal distribution the average of 4.6 is misleading. In fact over half of the reports received an average rating of 4.8 or higher. The distribution of the average report ratings is as shown:



It is clear from the high ratings that the laboratories place a high value on AFOSR's Summer Research Extension Programs.

3.0 SUBCONTRACTS SUMMARY

Table 1 provides a summary of the SREP subcontracts. The individual reports are published in volumes as shown:

Laboratory	Volume
Air Force Civil Engineering Laboratory	*
Armstrong Laboratory	1
Arnold Engineering Development Center	5
Frank J. Seiler Research Laboratory	5
Phillips Laboratory	2
Rome Laboratory	3
Wilford Hall Medical Center	5
Wright Laboratory	4A, 4B

*AFCEL was combined with Wright Laboratory's Flight Dynamics Directorate and Armstrong Laboratories Environics Directorate in 1993. All four of AFCEL's SREP awards went to Armstrong Laboratories Environics Directorate, and their reports are included with Armstrong Lab.

1993 SREP SUB-CONTRACT DATA

TABLE 1: SUBCONTRACTS SUMMARY

Report Author Author's University	Author's Degree	Sponsoring Lab	Performance Period		Contract Amount Univ. Cost Share
Abbott , Ben Electrical Engineering Vanderbilt University, Nashville, TN	M.S.	AEDC/	01/01/93	12/31/93	$19619.00 $0.00
Alberts , Thomas Mechanical Engineering Old Dominion University, Norfolk, VA	PhD	FJSRL/	01/01/93	04/15/94	$20000.00 $8000.00
Avula , Xavier Mechanical & Aerospace Engineering University of Missouri, Rolla, MO	PhD	AL/AO	01/01/93	04/15/94	$20000.00 $1836.00
Balakrishan , S. Mechanical & Aerospace Engineering University of Missouri, Rolla, MO	PhD	WL/MN	12/01/92	12/14/93	$20000.00 $3996.00
Baumgarten , Joseph Mechanical Engineering Iowa State University, Ames, IA	PhD	PL/VT	01/01/93	04/01/94	$19916.00 $9083.00
Bayard , Jean-Pierre Electrical & Electronic Engineering California State University, Sacramento, CA	PhD	RL/ER	01/01/93	12/31/93	$20000.00 $7423.00
Bellem , Raymond Electrical & Computer Engineering University of Arizona, Tucson, AZ	PhD	PL/VT	01/01/93	02/28/94	$19956.00 $0.00
Biegl , Csaba Electrical Engineering Vanderbilt University, Nashville, TN	PhD	AEDC/	01/01/93	12/31/93	$19999.00 $0.00
Biggs , Albert Electrical Engineering University of Alabama, Huntsville, AL	PhD	PL/WS	01/01/93	12/31/93	$19975.00 $0.00
Burkey , Theodore Chemistry Memphis State University, Memphis, TN	PhD	WL/MN	01/01/93	12/31/93	$20000.00 $18648.00
Burkholder , Robert Electrical Engineering Ohio State University, Columbus, OH	PhD	WL/AA	01/01/93	12/31/93	$20000.00 $6727.00
Callens, Jr. , Eugene Mechanical and Industrial Engineer Louisiana Tech University, Ruston, LA	PhD	WL/MN	01/01/93	12/31/93	$20000.00 $5700.00
Chapman , Gary Mechnical Engineering University of California, Berkeley, CA	PhD	WL/MN	01/01/93	12/31/94	$20000.00 $0.00
Chen , Chien-In Electrical Engineering Wright State University, Dayton, OH	PhD	WL/EL	01/01/93	12/31/93	$20000.00 $32065.00
Chen , Jer-sen Computer Science & Engineering Wright State University, Dayton, OH	PhD	AL/CF	01/01/93	12/31/93	$20000.00 $31763.00

1993 SREP SUB-CONTRACT DATA

Report Author Author's University	Author's Degree	Sponsoring Lab	Performance Period		Contract Amount Univ. Cost Share
Chen , Pinyuen Mathematics Syracuse University, Syracuse, NY	PhD	RL/IR	01/01/93	12/31/93	$20000.00 $0.00
Chow , Joe Industrial and Systems Engineering Florida International University, Miami, FL	PhD	WL/FI	01/01/93	01/14/94	$20000.00 $2500.00
Christensen , Thomas Physics University of Colorado, Colorado Springs, CO	PhD	FJSRL/	01/01/93	12/31/93	$20000.00 $5390.00
Collicott , Steven Aeronautics and Astronautical Engineering Purdue University, West Lafayette, IN	PhD	WL/FI	01/01/93	12/31/93	$20000.00 $13307.00
Cooke , Nancy Psychology New Mexico State University, Las Cruces, NM	PhD	AL/HR	01/01/93	12/31/93	$20000.00 $6178.00
Daley , Michael Electrical Engineering Memphis State, Memphis, TN	PhD	WHMC/	01/01/93	12/31/93	$20000.00 $18260.00
Davidson , Jennifer Electrical Engineering Iowa State University, Ames, IA	PhD	WL/MN	01/01/93	02/28/94	$19999.00 $0.00
Deivanayagam , Subramaniam Industrial Engineering Tennessee Technological University, Cookeville, TN	PhD	AL/HR	02/01/93	12/31/93	$20000.00 $12491.00
Elliott , David Engineering Arkansas Technology University, Russellville, AR	PhD	PL/RK	10/01/92	08/15/93	$20000.00 $50271.00
Erdman , Paul Physics and Astronomy University of Iowa, Iowa City, IA	M.S.	PL/RK	01/01/93	12/31/93	$20000.00 $26408.00
Ervin , Jamie Mechanical and Aerospace Engineering University of Dayton, Dayton, OH	PhD	WL/ML	01/01/93	12/31/93	$18632.00 $3000.00
Erwin , Daniel Aerospace Engineering University of Southern California, Los Angeles, CA	PhD	PL/RK	01/01/93	12/31/93	$19962.00 $12696.00
Ewert , Dan Electrical Engineering North Dakota State University, Fargo, ND	PhD	AL/AO	01/01/93	12/31/93	$20000.00 $2100.00
Farmer , Barry Materials Science and Engineering University of Virginia, Charlottesville, VA	PhD	WL/ML	01/01/93	02/28/94	$20000.00 $2000.00
Foo , Simon Electrical Engineering Florida State University, Tallahassee, FL	PhD	WL/MN	01/01/93	12/31/93	$19977.00 $0.00

1993 SREP SUB-CONTRACT DATA

Report Author Author's University	Author's Degree	Sponsoring Lab	Performance Period		Contract Amount Univ. Cost Share
Friedman , Jeffrey Physics University of Puerto Rico, Mayaguez, PR	PhD	PL/GP	01/01/93	12/31/93	$20000.00 $10233.00
Gerstman , Bernard Physics Florida International University, Miami, FL	PhD	AL/OE	01/01/93	04/30/94	$19947.00 $2443.00
Gillman , Ann Chemistry Eastern Kentucky University, Richmond, KY	M.S.	WL/PO	01/01/93	12/31/93	$20000.00 $15618.00
Gimmestad , Gary Research Institute Georgia Institute of Technology, Atlanta, GA	PhD	PL/LI	01/01/93	12/31/93	$20000.00 $0.00
Gould , Richard Mechanical and Aerospace Engineering North Carolina State University, Raleigh, NC	PhD	WL/PO	01/01/93	12/31/93	$20000.00 $8004.00
Graham , Gary Mechanical Engineering Ohio University, Athens, OH	PhD	WL/FI	01/01/93	12/31/93	$20000.00 $5497.00
Gramoll , Kurt Aerospace Engineering Georgia Institute of Technology, Atlanta, GA	PhD	AEDC/	01/01/93	12/31/93	$19707.00 $14552.00
Graul , Susan Chemistry Carnegie Mellon University, Pittsburgh, PA	PhD	PL/WS	01/01/93	03/31/94	$20000.00 $0.00
Griffin , Steven Engineering University of Texas, San Antonio, TX	M.S.	PL/VT	01/01/93	12/31/93	$20000.00 $0.00
Grubbs , Elmer Electrical Engineering New Mexico Highlands University, Las Vegas, NM	PhD	WL/AA	01/01/93	12/31/93	$20000.00 $6747.00
Gupta , Pushpa Mathematics University of Maine, Orono, ME	PhD	AL/AO	01/01/93	12/31/93	$20000.00 $1472.00
Hall , Ian Materials Science University of Delaware, Newark, DE	PhD	WL/ML	01/01/93	12/31/93	$20000.00 $9580.00
Hedman , Paul Chemical Engineering Brigham Young University, Provo, UT	PhD	WL/PO	01/01/93	12/31/93	$19999.00 $7755.00
Henry , Robert Electrical & Computer Engineering University of Southwestern Louisiana, Lafayette, LA	PhD	RL/C3	12/01/92	05/31/93	$19883.00 $11404.00
Henson , James Electrical Engineering University of Nevada, Reno, NV	PhD	PL/WS	01/01/93	12/31/93	$19913.00 $9338.00

1993 SREP SUB-CONTRACT DATA

Report Author Author's University	Author's Degree	Sponsoring Lab	Performance Period		Contract Amount Univ. Cost Share
Hoe , Benjamin Electrical Engineering Polytechnic University, Brooklyn, NY	M.S.	RL/C3	09/01/92	05/31/93	$19988.00 $7150.00
Hughes , Rod Psychology Bowling Green State University, Bowling Green, OH	M.S.	AL/CF	01/01/93	04/15/94	$20000.00 $20846.00
Hui , David Mechanical Engineering University of New Orleans, New Orleans, LA	PhD	WL/FI	01/01/93	12/31/93	$20000.00 $0.00
Humi , Mayer Mathematics Worcester Polytechnic Institut, Worcester, MA	PhD	PL/LI	01/01/93	12/31/93	$20000.00 $5000.00
Innocenti , Mario Aerospace Engineering Auburn University, Auburn, AL	PhD	WL/MN	01/01/93	02/28/94	$20000.00 $12536.00
Jean , Jack Computer Science & Engineering Wright State University, Dayton, OH	PhD	WL/AA	01/01/93	12/31/93	$20000.00 $34036.00
Jouny , Ismail Electrical Engineering Lafayette College, Easton, PA	PhD	WL/AA	01/01/93	12/31/93	$19381.00 $4500.00
Kaikhah , Khosrow Computer Science Southwest Texas State College, San Marcos, TX	PhD	RL/IR	01/01/93	12/31/93	$20000.00 $0.00
Kaw , Autar Mechanical Engineering University of South Florida, Tampa, FL	PhD	WL/ML	01/01/93	12/31/93	$20000.00 $22556.00
Kheyfets , Arkady Mathematics North Carolina State University, Raleigh, NC	PhD	PL/LI	01/01/93	12/31/93	$20000.00 $2500.00
Kitchart , Mark Mechanical Engineering North Carolina A & T State University, Greensboro, NC	M.S.	AL/EQ	01/01/93	12/31/93	$20000.00 $0.00
Koblasz , Arthur Civil Engineering Georgia Institute of Technology, Atlanta, GA	PhD	AL/AO	01/01/93	12/31/93	$19826.00 $0.00
Koivo , A. Electrical Engineering Purdue University, West Lafayette, IN	PhD	AL/CF	01/01/93	06/30/94	$20000.00 $0.00
Kundich , Robert Biomedical Engineering University of Tennessee, Memphis, TN	PhD	AL/CF	01/01/93	12/31/94	$20000.00 $23045.00
Kuo , Spencer Electrical Engineering Polytechnic University, Farmingdale, NY	PhD	PL/GP	01/01/93	04/30/94	$20000.00 $9731.00

1993 SREP SUB-CONTRACT DATA

Report Author Author's University	Author's Degree	Sponsoring Lab	Performance Period		Contract Amount Univ. Cost Share
Liou , Juin Electrical and Computer Engineering Universtiy of Central Florida, Orlando, FL	PhD	WL/EL	01/01/93	12/31/93	$20000.00 $9073.00
Liou , Shy-Shenq Engineering San Francisco State Univesity, San Francisco, CA	PhD	WL/PO	01/01/93	12/31/93	$20000.00 $13387.00
Manoranjan , Valipuram Pure and Applied Mathematics Washington State University, Pullman, WA	PhD	AL/EQ	01/01/93	12/31/93	$19956.00 $10041.00
Marks , Dallas Electrical and Computer Engineering University of Cincinnati M.L., Cincinnati, OH	M.S.	WL/AA	10/01/92	06/30/93	$20000.00 $4731.00
Monsay , Evelyn Physics Le Moyne College, Syracuse, NY	PhD	RL/OC	01/01/93	12/31/93	$19634.00 $1510.00
Moor , William Industrial & Management Engineering Arizona State University, Tempe, AZ	PhD	AL/HR	01/01/93	12/31/93	$20000.00 $4833.00
Moore , Carlyle Physics Morehouse College, Atlanta, GA	PhD	AEDC/	01/01/93	12/31/93	$20000.00 $4880.00
Mulligan , B. Psychology University of Georgia Research, Athens, GA	PhD	AL/OE	01/01/93	04/15/94	$19998.00 $13936.00
Murphy , Richard Physics University of Missouri, Kansas City, MO	PhD	PL/WS	01/01/93	12/31/93	$20000.00 $13022.00
Nilan , Michael Information Studies Syracuse University, Syracuse, NY	PhD	RL/C3	01/01/93	12/31/93	$19998.00 $13016.00
Parrish , Allen Computer Science University of Alabama, Tuscaloosa, AL	PhD	RL/C3	01/01/93	12/31/93	$19919.00 $20599.00
Piersma , Bernard Chemistry Houghton College, Houghton, NY	PhD	FJSRL/	01/01/93	12/31/93	$20000.00 $4000.00
Potasek , Mary Applied Physics Columbia University, New York, NY	PhD	WL/ML	12/01/93	11/30/93	$20000.00 $7806.00
Qazi , Salahuddin Optical Communications SUNY/Institute of Technology, Utica, NY	PhD	RL/OC	01/01/93	12/31/93	$20000.00 $68000.00
Reardon , Kenneth Agricultural and Chemical Engineering Colorado State University, Fort Collins, CO	PhD	AL/EQ	01/01/93	01/31/94	$19996.00 $12561.00

1993 SREP SUB-CONTRACT DATA

Report Author Author's University	Author's Degree	Sponsoring Lab	Performance Period		Contract Amount Univ. Cost Share
Reynolds , David Biomedical & Human Factors Wright State University, Dayton, OH	PhD	AL/CF	01/01/93	06/30/94	$20000.00 $14063.00
Robinson , Donald Chemistry Xavier University of Louisiana, New Orleans, LA	PhD	AL/OE	01/01/93	06/30/94	$20000.00 $12935.00
Rodriguez , Armando Electrical Engineering Arizona State University, Tempe, AZ	PhD	WL/MN	01/01/93	12/31/93	$20000.00 $0.00
Roe , Larry Mechanical Engineering Virginia Polytechnic Inst & State Coll., Blacksburg, VA	PhD	WL/PO	01/01/93	12/31/93	$20000.00 $11421.00
Romeu , Jorge Assistant Prof. of Mathematics SUNY College at Cortland, Cortland, NY	PhD	RL/OC	01/01/93	12/31/93	$19997.00 $7129.00
Roppel , Thaddeus Electrical Engineering Auburn University, Auburn, AL	PhD	WL/MN	01/01/93	12/31/93	$20000.00 $21133.00
Roznowski , Mary Psychology Ohio State University, Columbus, OH	PhD	AL/HR	01/01/93	03/31/94	$19953.00 $6086.00
Rudzinski , Walter Chemistry Southwest Texas State University, San Marcos, TX	PhD	AL/OE	01/01/93	12/31/93	$20000.00 $10120.00
Sargent , Robert Engineering and Computer Science Syracuse University, Syracuse, NY	PhD	RL/XP	01/01/93	12/31/93	$20000.00 $11931.00
Schonberg , William Civil and Environmental Engineering University of Alabama, Huntsville, AL	PhD	WL/MN	01/01/93	12/31/93	$19991.00 $5083.00
Shaw , Arnab Electrical Engineering Wright State University, Dayton, OH	PhD	WL/AA	01/01/93	12/31/93	$20000.00 $4766.00
Shively , Jon Engineering & Computer Science California State University, Northridge, CA	PhD	PL/VT	01/01/93	12/31/93	$20000.00 $9782.00
Slater , Robert Mechanical & Industrial Engineering University of Cincinnati, Cincinnati, OH	M.S.	WL/FI	01/01/93	12/31/93	$20000.00 $8257.00
Stenzel , Johanna Arts & Sciences University of Houston, Victoria, TX	PhD	PL/LI	01/01/93	12/31/93	$20000.00 $9056.00
Tan , Arjun Physics Alabama A & M University, Normal, AL	PhD	PL/WS	01/01/93	12/31/93	$20000.00 $1000.00

1993 SREP SUB-CONTRACT DATA

Report Author Author's University	Author's Degree	Sponsoring Lab	Performance Period		Contract Amount Univ. Cost Share
Tetrick , Lois Industrial Relations Prog Wayne State University, Detroit, MI	PhD	AL/HR	01/01/93	12/31/93	$20000.00 $17872.00
Tew , Jeffery Industrial & Systems Engineering Virginia Polytechnic Institute, Blacksburg, VA	PhD	RL/IR	05/31/93	12/31/93	$16489.00 $4546.00
Tribikram , Kundu Civil Engineering and Engineering Universtiy of Arizona, Tucson, AZ	PhD	WL/ML	01/01/93	12/31/93	$20000.00 $9685.00
Tuthill , Theresa Electrical Engineering University of Dayton, Dayton, OH	PhD	WL/ML	01/01/93	12/31/93	$20000.00 $24002.00
Venkatasubraman , Ramasubrama Electrical and Computer Engineering University of Nevada, Las Vegas, NV	PhD	WL/ML	01/01/93	12/31/93	$20000.00 $18776.00
Wang , Xingwu Electrical Engineering Alfred University, Alfred, NY	PhD	AL/EQ	01/01/93	12/31/93	$20000.00 $10000.00
Whitefield , Philip Physics University of Missouri, Rolla, MO	PhD	PL/LI	01/01/93	03/01/94	$20000.00 $11040.00
Wightman , Colin Electrical Engineering New Mexico Institute of Mining, Socorro, NM	PhD	RL/IR	01/01/93	12/31/93	$20000.00 $1850.00
Womack , Michael Natural Science and Mathematics Macon College, Macon, GA	PhD	AL/OE	01/01/93	06/30/94	$19028.00 $6066.00
Yuvarajan , Subbaraya Electrical Engineering North Dakota State University, Fargo, ND	PhD	WL/PO	01/01/93	12/31/93	$19985.00 $22974.00

APPENDIX 1:

SAMPLE SREP SUBCONTRACT

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
1993 SUMMER RESEARCH EXTENSION PROGRAM SUBCONTRACT 93-133

BETWEEN

Research & Development Laboratories
5800 Uplander Way
Culver City, CA 90230-6608

AND

San Francisco State University
University Comptroller
San Francisco, CA 94132

REFERENCE: Summer Research Extension Program Proposal 93-133
Start Date: 01/01/93 End Date: 12/31/93
Proposal Amount: $20,000.00

(1) PRINCIPAL INVESTIGATOR: Dr. Shy Shenq P. Liou
Engineering
San Francisco State University
San Francisco, CA 94132

(2) UNITED STATES AFOSR CONTRACT NUMBER: F49620-90-C-09076

(3) CATALOG OF FEDERAL DOMESTIC ASSISTANCE NUMBER (CFDA): 12.800
PROJECT TITLE: AIR FORCE DEFENSE RESEARCH SOURCES PROGRAM

(4) ATTACHMENTS 1 AND 2: SREP REPORT INSTRUCTIONS

*** SIGN SREP SUBCONTRACT AND RETURN TO RDL***

1. BACKGROUND: Research & Development Laboratories (RDL) is under contract (F49620-90-C-0076) to the United States Air Force to administer the Summer Research Programs (SRP), sponsored by the Air Force Office of Scientific Research (AFOSR), Bolling Air Force Base, D.C. Under the SRP, a selected number of college faculty members and graduate students spend part of the summer conducting research in Air Force laboratories. After completion of the summer tour participants may submit, through their home institutions, proposals for follow-on research. The follow-on research is known as the Summer Research Extension Program (SREP). Approximately 75 SREP proposals annually will be selected by the Air Force for funding of up to $20,000; shared funding by the academic institution is encouraged. SREP efforts selected for funding are administered by RDL through subcontracts with the institutions. This subcontract represents such an agreement between RDL and the institution designated in Section 5 below.

2. RDL PAYMENTS: RDL will provide the following payments to SREP institutions:

- 90 percent of the negotiated SREP dollar amount at the start of the SREP Research period.
- the remainder of the funds within 30 days after receipt at RDL of the acceptable written final report for the SREP research.

3. INSTITUTION'S RESPONSIBILITIES: As a subcontractor to RDL, the institution designated on the title page will:

a. Assure that the research performed and the resources utilized adhere to those defined in the SREP proposal.
b. Provide the level and amounts of institutional support specified in the RIP proposal.
c. Notify RDL as soon as possible, but not later than 30 days, of any changes in 3a or 3b above, or any change to the assignment or amount of participation of the Principal Investigator designated on the title page.

d. Assure that the research is completed and the final report is delivered to RDL not later than twelve months from the effective date of this subcontract, but no later than December 31, 1993. The effective date of the subcontract is one week after the date that the institution's contracting representative signs this subcontract, but no later than January 15, 1993.

e. Assure that the final report is submitted in accordance with Attachment 1.

f. Agree that any release of information relating to this subcontract (news releases, articles, manuscripts, brochures, advertisements, still and motion pictures, speeches, trade association meetings, symposia, etc.) will include a statement that the project or effort depicted was or is sponsored by: Air Force Office of Scientific Research, Bolling AFB, D.C.

g. Notify RDL of inventions or patents claimed as the result of this research as specified in Attachment 1.

h. RDL is required by the prime contract to flow down patent rights and technical data requirements in this subcontract. Attachment 2 to this subcontract contains a list of contract clauses incorporated by reference in the prime contract.

4. All notices to RDL shall be addressed to:

RDL Summer Research Program Office
5800 Uplander Way
Culver City, CA 90230-6608

5. By their signatures below, the parties agree to the provisions of this subcontract.

Abe S. Sopher
RDL Contracts Manager

Signature of Institution Contracting Official

Typed/Printed Name

Date

Title

Institution

(Date/Phone)

ATTACHMENT 2
CONTRACT CLAUSES

This contract incorporates by reference the following clauses of the Federal Acquisition Regulations (FAR), with the same force and effect as if they were given in full text. Upon request, the Contracting Officer or RDL will make their full text available (FAR 52.252-2).

FAR CLAUSES	TITLE AND DATE
52.202-1	DEFINITIONS (SEP 1991)
52.203-1	OFFICIALS NOT TO BENEFIT (APR 1984)
52.203-3	GRATUITIES (APR 1984)
52.203-5	COVENANT AGAINST CONTINGENT FEES (APR 1984)
52.304-6	RESTRICTIONS ON SUBCONTRACTOR SALES TO THE GOVERNMENT (JUL 1985)
52.203-7	ANTI-KICKBACK PROCEDURES (OCT 1988)
52.203-12	LIMITATION ON PAYMENTS TO INFLUENCE CERTAIN FEDERAL TRANSACTIONS (JAN 1990)
52.204-2	SECURITY REQUIREMENTS (APR 1984)
52.209-6	PROTECTING THE GOVERNMENT'S INTEREST WHEN SUBCONTRACTING WITH CONTRACTORS DEBARRED, SUSPENDED, OR PROPOSED FOR DEBARMENT (NOV 1992)
52.212-8	DEFENSE PRIORITY AND ALLOCATION REQUIREMENTS (SEP 1990)
52.215-1	EXAMINATION OF RECORDS BY COMPTROLLER GENERAL (APR 1984)
52.215-2	AUDIT - NEGOTIATION (DEC 1989)
52.222-26	EQUAL OPPORTUNITY (APR 1984)
52.222-28	EQUAL OPPORTUNITY PREAWARD CLEARANCE OF SUBCONTRACTS (APR 1984)

52.222-35	AFFIRMATIVE ACTION FOR SPECIAL DISABLED AND VIETNAM ERA VETERANS (APR 1984)
52.222-36	AFFIRMATIVE ACTION FOR HANDICAPPED WORKERS (APR 1984)
52.222-37	EMPLOYMENT REPORTS ON SPECIAL DISABLED VETERAN AND VETERANS OF THE VIETNAM ERA (JAN 1988)
52.223-2	CLEAN AIR AND WATER (APR 1984)
52.232-6	DRUG-FREE WORKPLACE (JUL 1990)
52.224-1	PRIVACY ACT NOTIFICATION (APR 1984)
52.224-2	PRIVACY ACT (APR 1984)
52.225-13	RESTRICTIONS ON CONTRACTING WITH SANCTIONED PERSONS (MAY 1989)
52.227-1	AUTHORIZATION AND CONSENT (APR 1984)
52.227-2	NOTICE AND ASSISTANCE REGARDING PATENT AND COPYRIGHT INFRINGEMENT (APR 1984)
52.227-10	FILING OF PATENT APPLICATIONS - CLASSIFIED SUBJECT MATTER (APR 1984)
52.227-11	PATENT RIGHTS - RETENTION BY THE CONTRACTOR (SHORT FORM) (JUN 1989)
52.228-6	INSURANCE - IMMUNITY FROM TORT LIABILITY (APR 1984)
52.228-7	INSURANCE - LIABILITY TO THIRD PERSONS (APR 1984)
52.230-5	DISCLOSURE AND CONSISTENCY OF COST ACCOUNTING PRACTICES (AUG 1992)
52.232-23	ASSIGNMENT OF CLAIMS (JAN 1986)
52.237-3	CONTINUITY OF SERVICES (JAN 1991)

52.246-25	LIMITATION OF LIABILITY - SERVICES (APR 1984)
52.249-6	TERMINATION (COST-REIMBURSEMENT) (MAY 1986)
52.249-14	EXCUSABLE DELAYS (APR 1984)
52.251-1	GOVERNMENT SUPPLY SOURCES (APR 1984)

APPENDIX 2:

SAMPLE TECHNICAL EVALUATION FORM

1993 SUMMER RESEARCH EXTENSION PROGRAM

RIP NO.: 93-0092
RIP ASSOCIATE: Dr. Gary T. Chapman

Provided are several evaluation statements followed by ratings of (1) through (5). A rating of (1) is the lowest and (5) is the highest. Circle the rating level number you best feel rates the statement. Document additional comments on the back of this evaluation form.

Mail or fax the completed form to :

RDL
Attn: 1993 SREP TECH EVALS
5800 Uplander Way
Culver City, CA 90230-6608
(FAX: 310 216-5940)

1.	This SREP report has a high level of technical merit.	1	2	3	4	5
2.	The SREP program is important to accomplishing the labs's mission	1	2	3	4	5
3.	This SREP report accomplished what the associate's proposal promised.	1	2	3	4	5
4.	This SREP report addresses area(s) important to the USAF	1	2	3	4	5
5.	The USAF should continue to pursue the research in this SREP report	1	2	3	4	5
6.	The USAF should maintain research relationships with this SREP associate	1	2	3	4	5
7.	The money spent on this SREP effort was well worth it	1	2	3	4	5
8.	This SREP report is well organized and well written	1	2	3	4	5
9.	I'll be eager to be a focal point for summer and SREP associates in the future.	1	2	3	4	5
10.	The one-year period for complete SREP research is about right	1	2	3	4	5

******USE THE BACK OF THIS FORM FOR ADDITIONAL COMMENTS******

LAB FOCAL POINT'S NAME (PRINT): _____

OFFICE SYMBOL: _____ PHONE: _____

SCAN PERFORMANCE OF INFINITE ARRAYS OF MICROSTRIP-FED DIPOLES WITH BENT ARMS PRINTED ON PROTRUDING SUBSTRATES

Jean-Pierre R. Bayard
Associate Professor
Department of Electrical & Electronic Engineering
California State University, Sacramento
6000 J Street, Sacramento, CA 95819-6019, U.S.A

Final Report for:
Summer Research Extension Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, D.C.

and

California State University, Sacramento

December 1993

SCAN PERFORMANCE OF INFINITE ARRAYS OF MICROSTRIP-FED DIPOLES WITH BENT ARMS PRINTED ON PROTRUDING SUBSTRATES

Jean-Pierre R. Bayard
Associate Professor
Department of Electrical & Electronic Engineering
California State University, Sacramento

Abstract

Scan results are presented for infinite arrays of microstrip-fed dipoles printed on protruding dielectric substrates. Active impedance and cross polarization values are calculated for dipoles with bent arms, and for the array with a near-field dielectric radome cover. It is found that bending the dipole's arms produces a small improvement in the scan range as compared to the element with the arms straight, and that a dielectric radome with a low permittivity causes a small shift in the impedance and cross polarization curves. The scan limitation of the array is however dictated by the presence of the coplanar feedlines with some improvement achievable by varying the substrate's thickness and permittivity, as well as the dipole/feed geometry. A dipole element with a modified coplanar feed design is introduced for wider scan coverage.

SCAN PERFORMANCE OF INFINITE ARRAYS OF MICROSTRIP-FED DIPOLES WITH BENT ARMS PRINTED ON PROTRUDING SUBSTRATES

Jean-Pierre R. Bayard

Introduction

Recently, antennas printed on dielectric substrates that protrude a finite height from a ground plane have generated considerable interest in radar and communication applications. Such element is fast becoming a very attractive candidate for monolithic phased array technology for a variety of reasons. First and foremost, active and passive circuits can be built on the substrate extension (with a potentially high dielectric constant) behind the ground plane, isolated from the radiating elements. The configuration is also modular, permitting easy replacement of a defective column of antenna elements. Thirdly, heat typically generated in an antenna /electronic module is easier extracted from the present configuration than from their layered counterparts such as those with microstrip elements at the front end.

In [1], a hybrid moment method solution was presented for an infinite array of center-fed dipoles printed parallel to the ground plane. Then in [2-4], the solution was extended to handle electric currents parallel and perpendicular to the ground plane, printed on both sides of the substrate, and the presence of a dielectric radome. The accuracy of the method was exhibited by comparing its results with measurements, as in [1], and by predicting, as have other workers in [5,6], feed-induced blind spots in the scan coverage of dipole arrays. The present effort is yet another extension/application of the solution presented in [1] for the dipole array. The schematic of the array unit cell is shown in Figure 1. The element geometry is that of a dipole with bent arms fed by coplanar stubs electromagnetically coupled to a microstrip transmission line printed on the other side of the substrate (dotted line in Figure 2). In order to analyze the element depicted in Figure 2, the numerical solution is required to model current distributions defined on linearly-tapered domains, in addition to treating electric currents that are printed parallel, perpendicular to the ground plane and on both sides of the substrate. The applicability of the method is not confined to dipole elements alone; it is indeed capable of modeling many other types of geometries such as the printed slot antenna with linear taper.

Following a brief outline of the method emphasizing the areas which differ from [4], results will be shown for the array, in particular, values for the active impedance and cross polarization are plotted for E-plane scan. It is anticipated, as suggested in [6] for arrays of dipoles in free-space, that the inclination of the arms may reduce the effects of the feed on scan, but this improvement may be accompanied with some polarization degradation. These potential effects as well as the effects of the substrate and radome parameters on the array scan range will be demonstrated. Lastly, the element shown in Figure 3 is considered as a possible

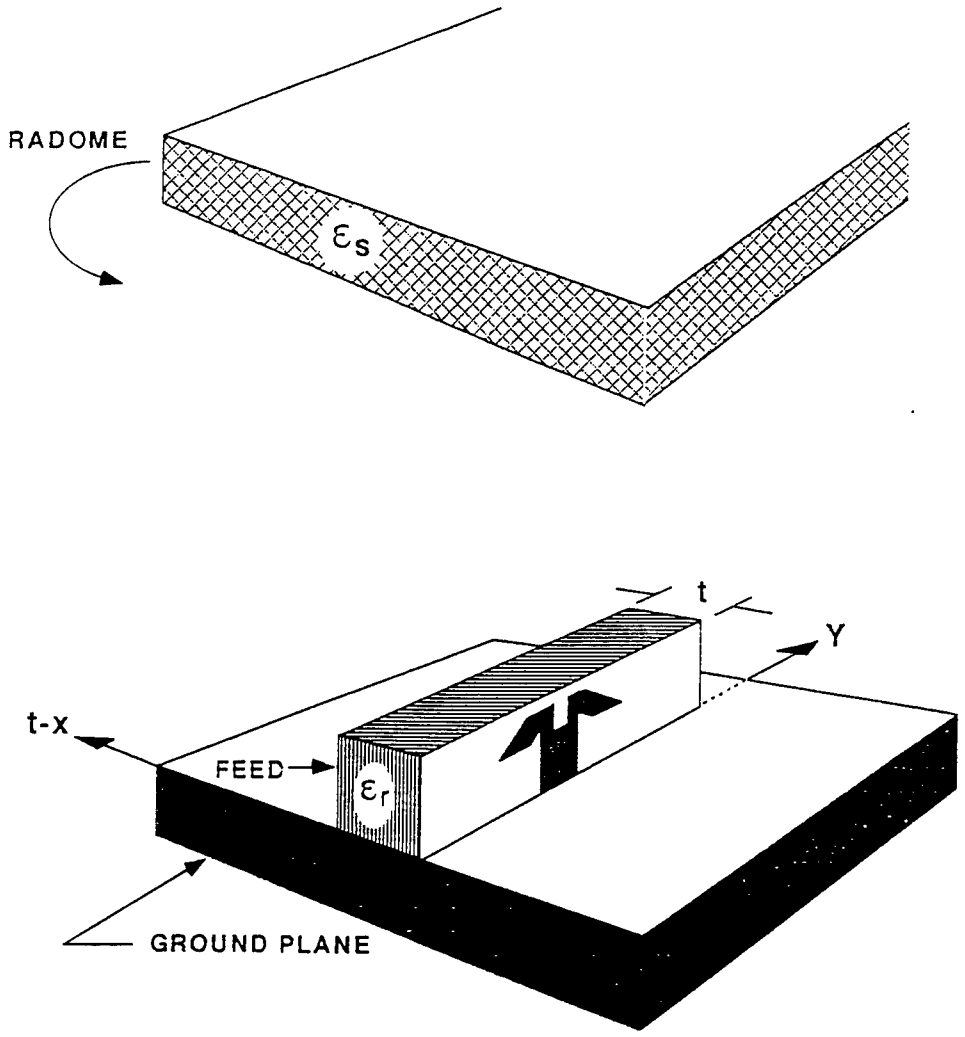


Figure 1. The unit cell of the array.

L_α	0.16λ
W_α	0.05715λ
W_c	0.02λ
W_b	0.20λ
L_f	0.255λ
W_f	0.010λ
Y_f	0.0675λ
L_c	0.10λ
L_1	0.065λ
L_2	0.045λ
L_s	0.25λ
θ_α	48.8°

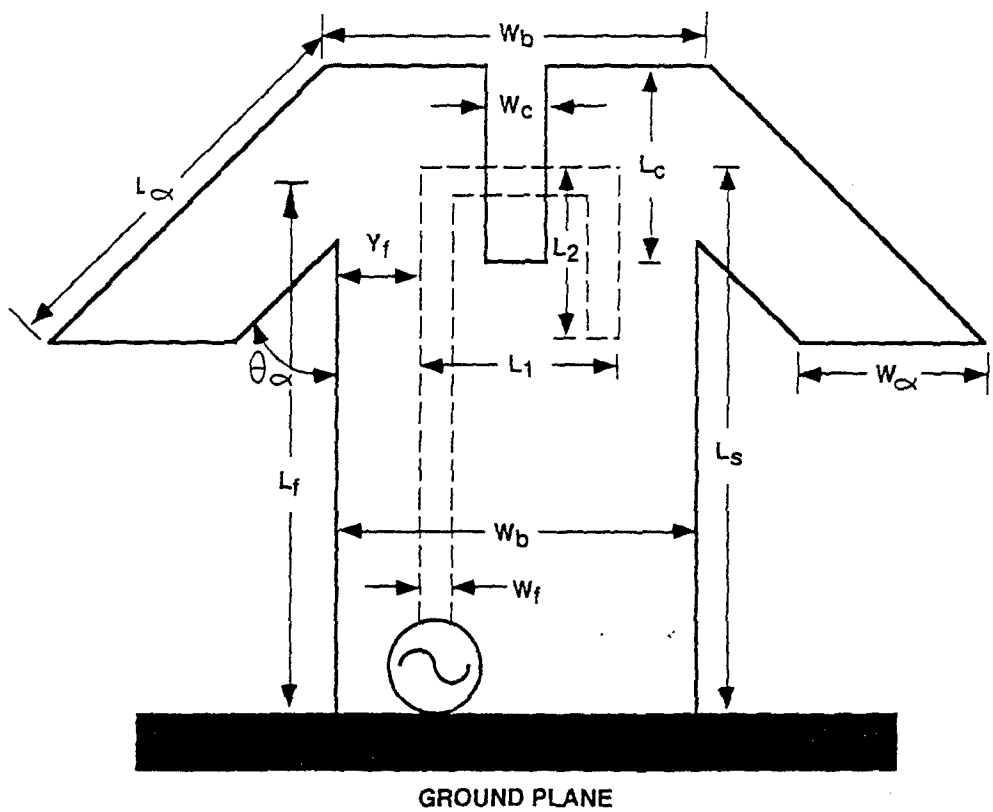


Figure 2. the geometry of the dipole element fed by coplanar stubs proximity coupled to a microstrip line.

L_α	0.16λ
W_α	0.05715λ
W_c	0.02λ
W_b	0.20λ
L_f	0.255λ
W_f	0.010λ
Y_f	0.0675λ
L_c	0.10λ
L_1	0.065λ
L_2	0.045λ
L_s	0.25λ
$\theta\alpha$	48.8°

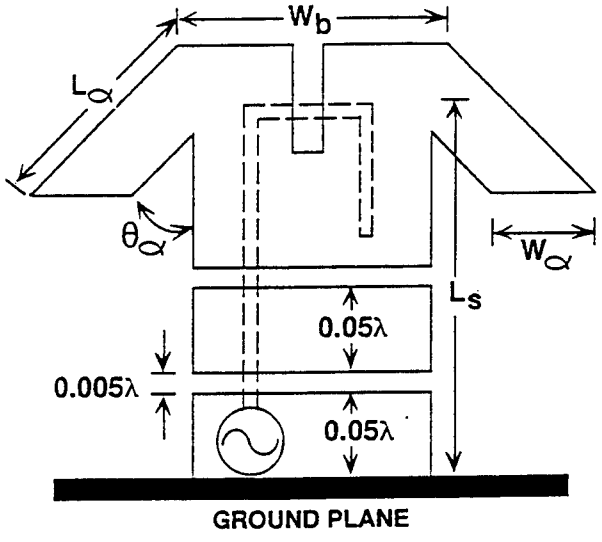


Figure 3. A dipole element with a modified feed configuration.

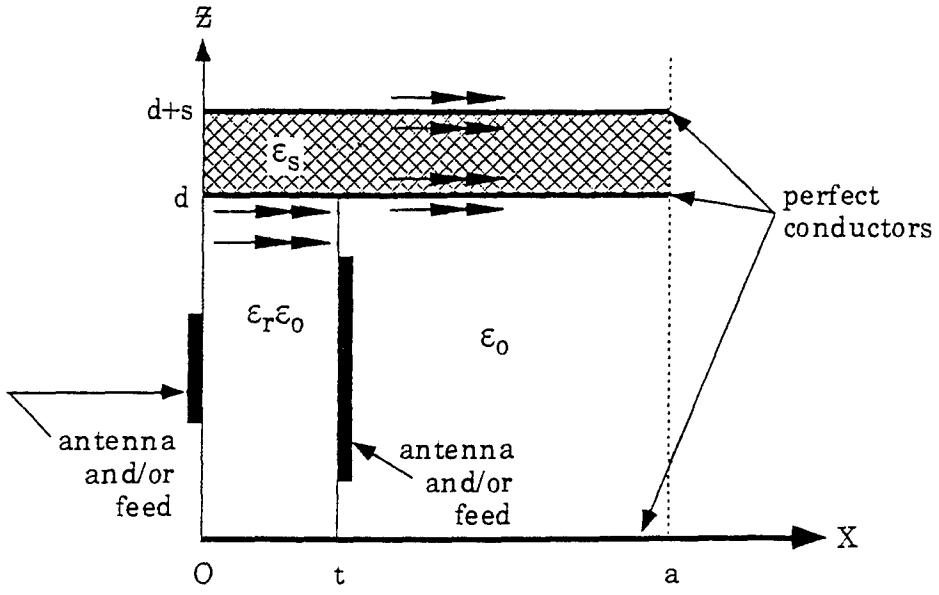
avenue for achieving wider scan coverage.

SUMMARY

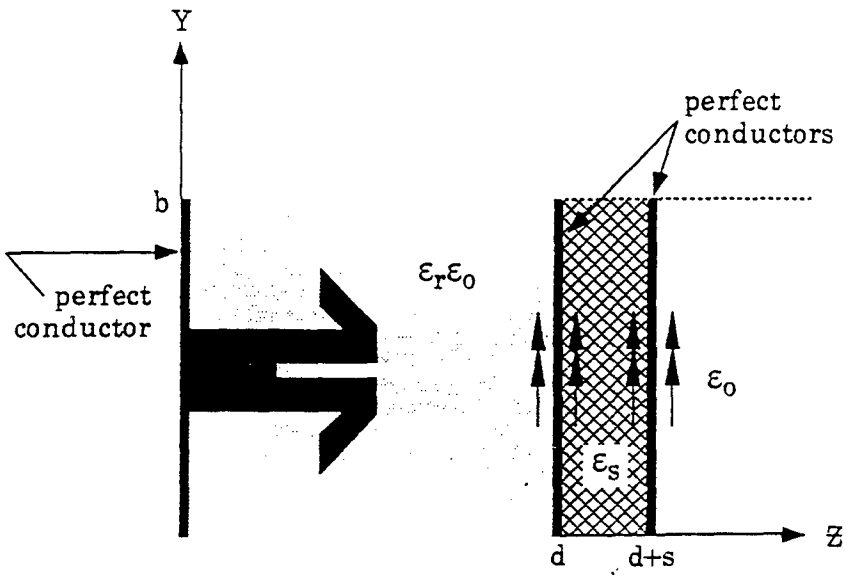
Consider the array whose unit cell appears in Figure 1. It is our objective to calculate the element active impedance caused by a delta-gap generator located on a y or z-directed rectangular mode, or the reflection coefficient caused by a plane wave illuminating the face of the array. As shown in Figure 4, the geometry, which exhibits non-planar characteristics for $z \leq d$, is transformed by applying the equivalence principle at z=d and at z=d+s, and by inserting perfect conductors at these locations. In doing so, the array unit cell is divided into three regions with equivalent magnetic sources at the boundaries: a) An inhomogeneously-filled parallel-plate region (z<d) with the antenna/feed currents at x=0 and x=t, and with equivalent sources at z=d; b) a planar and homogeneous dielectric (d<z<d+s) with equivalent sources at both interfaces; c) a semi-infinite free-space region (z>d+s) with equivalent sources on its boundary. We formulate the fields in each region separately, and apply the necessary boundary conditions required for uniqueness of the solution.

For the free-space and for the radome regions, the electromagnetic fields are expressed via well-known Floquet type basis functions with unknown coefficients. Those in the free-space region are propagating away from the face of the array, whereas those inside the radome region have standing wave characteristics. For the parallel-plate region, through superposition, the fields caused by the electric currents of the antenna and feed and those produced by the equivalent sources at z=d are formulated separately. The contributions from the electric currents are found by convolving the unknown distribution, e.g., J(x,y,z), where x is either 0 or t, with the Green's functions for infinitesimal current sources located inside the guide at x=0 or x=t. The contributions from the equivalent sources are expressed as summations of the so-called LSM and LSE modes (see [4]) similar to those used in [7]. At this time, we apply the electromagnetic boundary conditions at z=d, d+s, and on the surface of the conductors at x=0 and x=t. These conditions are expressed in matrix form by testing those at z=d and z=d+s with the conjugate of the Floquet functions, and by using a Galerkin procedure for those at x=0 and x=t. The part of the method concerning the conditions at x=0 and x=t represents the essence of the extension, thus warranting a somewhat more detailed discussion.

In order to implement the conditions that the electric field components tangential to the radiating conductors at x=0,t vanish, J(x,y,z) is approximated by finite sums of selected basis functions:



(a) X - Z plane



(b) Y - Z plane

Figure 4. The equivalent model for the unit cell.

$$\vec{J}(y,z)=\vec{J}_y+\vec{J}_z+\vec{J}_t+\vec{J}_\gamma$$

where

$$\vec{J}_y=\sum_{n=1}^{MPY}\hat{y}\ a_n\frac{sink_o(h_n-|y-y_n|)}{W_n\sin(k_oh_n)},$$

for $k_o=\omega\sqrt{\mu_o\epsilon_o}$, W_n *is the mode width,* $2h_n$ *is the mode length*
$z_n-W_n/2\le z\le z_n+W_n/2,\ y_n-h_n\le y\le y_n+h_n,$

$$\vec{J}_z=\sum_{m=1}^{MPZ}\hat{z}\ b_m\frac{sink_o(h_m-|z-z_m|)}{W_m\sin(k_oh_m)},$$

for $y_m-W_m/2\le y\le y_m+W_m/2,\ z_m-h_m\le z\le z_m+h_m,$

$$\vec{J}_t=\sum_{p=1}^{MPT}\hat{y}\ c_p\frac{sink_o(h_p^{\pm}-|y-y'(z)|)}{W_p\sin(k_oh_p^{\pm})}$$

for $(z-z_p)\tan\gamma_p-h_p^-\le y-y_p\le(z-z_p)\tan\gamma_p+h_p^+,\ z_p-W_p/2\le z\le z_p+W_p/2,$
$y'(z)=y_p+(z-z_p)\tan\gamma_p,$

$$\vec{J}_\gamma=\sum_{q=1}^{MPG}d_q(\hat{y}\sin\gamma^{\pm}+\hat{z}\cos\gamma^{\pm})J_\gamma^{\pm}(y,z),$$

with $J_\gamma^{\pm}(y,z)=\dfrac{sink_o(h_q^{\pm}-|z-z_q|))}{W_q\sin(k_oh_q^{\pm})}$, $z_q-h_q^-\le z\le z_q+h_q^+$, *and*
$(z-z_q)\tan\gamma_q^{\pm}-W_q/2\le y-y_q\le(z-z_q)\tan\gamma_q^{\pm}+W_q/2.$

(1)

a_n, b_m, c_p and d_q are as yet undetermined expansion coefficients. First, realize that the dimensions and locations of the modes are adjustable individually offering great deal of generality in the shapes that the above distribution can model. The first two sums in equation (1) contain standard y and z-directed piecewise sinusoidal functions defined on rectangular domains (see Figure 5). For the z-directed modes with their centers at $z_m=0$ only their upper-half portions are considered ($0\le z\le h_m$) since the current does not vanish at the antenna/ground plane junction. The other two sums in equation (1) contain modes that are defined on tapered domains as described in [8]. In the third sum, there are y-directed modes, but defined on a tilted domain (see Figure 6). The fourth sum contains two linearly-tapered sinusoidal halves, each with its own taper angle and length (see Figure 6). Such a mode (γ-directed), in addition to modeling current distribution on a tapered domain, is capable of treating bents maintaining current continuity as the current changes direction. Note that for $\gamma=0$ the tilted y-directed mode becomes a rectangular y-directed mode, and that for $\gamma^+=\gamma^-=0$, the γ-directed mode becomes a rectangular z-directed mode. Also $\gamma=90$ and γ^- or $\gamma^+=90$ are not acceptable values, but these conditions can be readily treated with the rectangular modes. The implementation of these tapered modes in the moment method solution of [4] requires changing the limits of the expansion and testing integrals for the tilted y-directed modes, and introducing a new set of boundary condition, i.e., $E_\gamma=0$, on the tapered domains of the γ-directed modes. While these

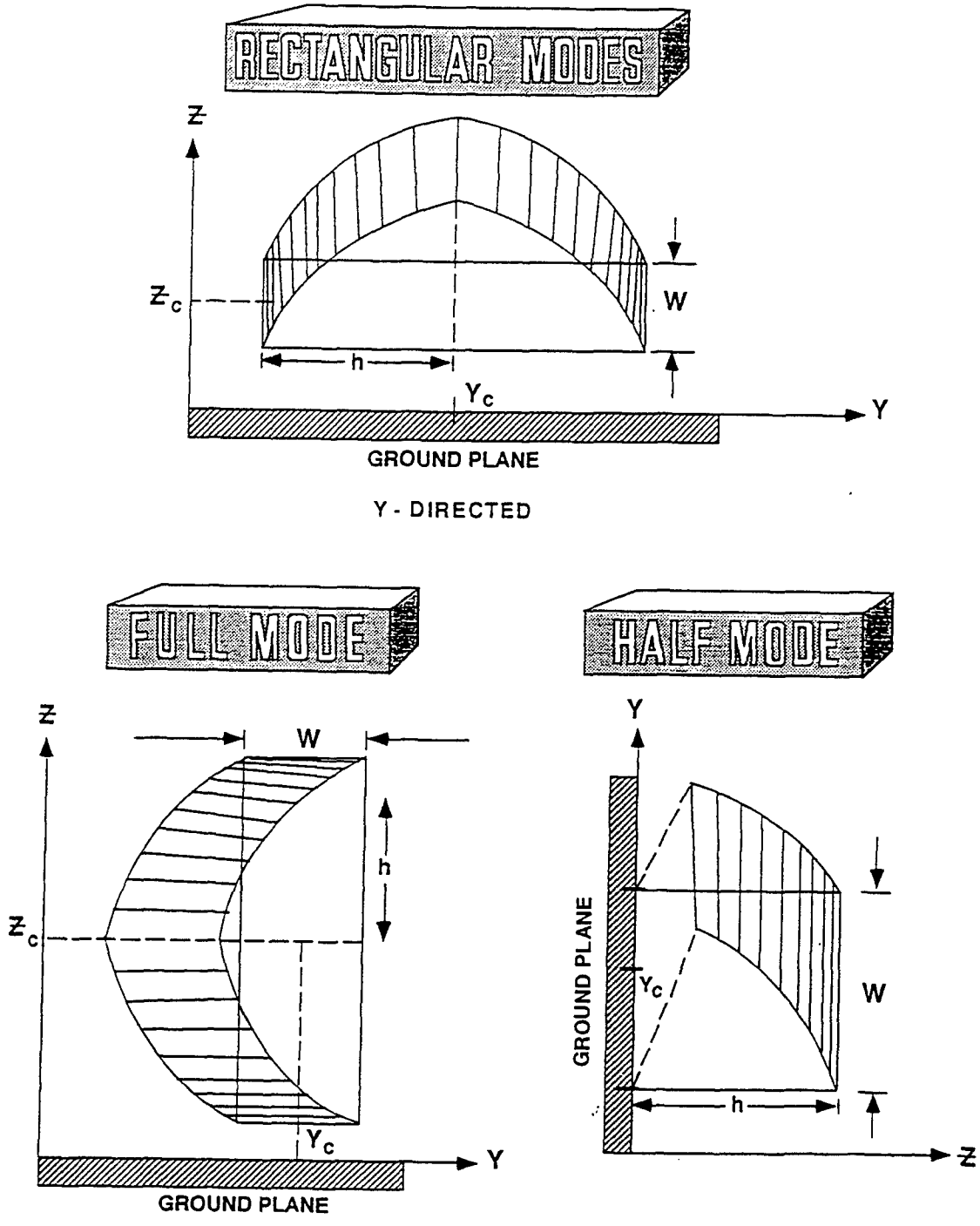


Figure 5. The rectangular modes used to expand the electric current distribution.

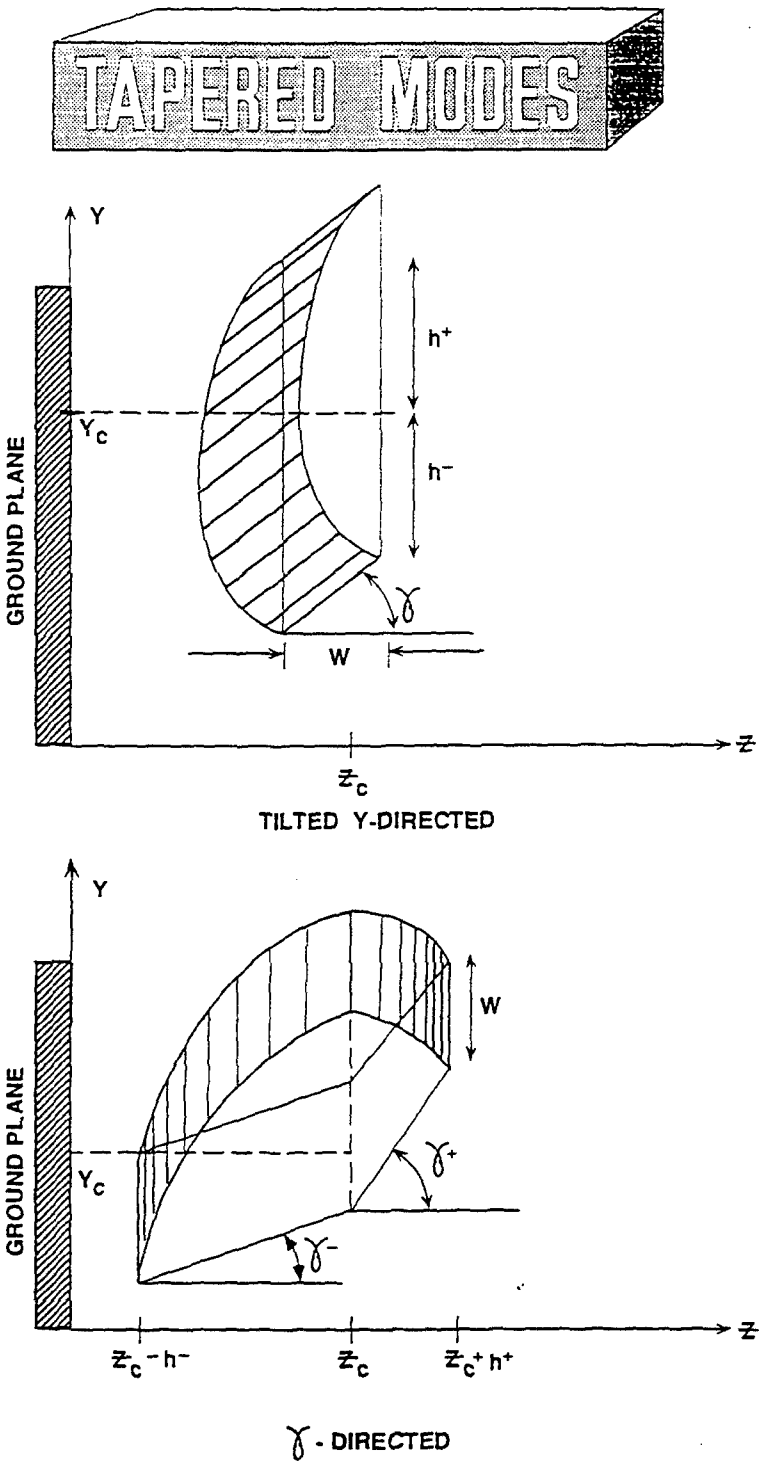


Figure 6. The tapered modes used to expand the electric current distribution.

integrations are all carried out analytically, the resulting expressions are fairly complex, and their numerical evaluations are more computer-intensive than those involving the rectangular modes. For that reason, tapered modes are only used to model tapered regions.

RESULTS

Experimental Verification

To verify the theoretical and coding extensions, numerical results and waveguide simulator data are compared for two slot elements (see Figures 7 and 8). In Figure 7, the reflection coefficient for an array of linearly-tapered slot elements in free-space is calculated and compared with data published in [8]. Given that the slot element is lossless ($|\Gamma|=1$ in the calculations), the agreement between theory and experiment is quite good. In Figure 8, the exponentially-tapered slot element of [9] loaded with 100Ω is considered. The exponential taper is approximated in the calculations by linear segments as demonstrated in Figure 8. Here too, good agreement exists between the calculations and the measurements of [9]. Although these arrays are both standing in free-space, the results provide confidence in the code's capability of modeling tapered geometries, which is the focus of the analytical part of this work.

Numerical Calculations

For the numerical study, first consider the geometry shown in Figure 2. The delta-gap generator is relocated to the midpoint of the y-directed segment of the microstrip line. Given an ideal transmission line, this would be equivalent to referencing the active impedance to the dipole's apex. In practice, this can be accomplished by extending the microstrip line beyond the ground plane. In order to approximate the electric currents on the dipole and on the feed structure, 97 expansion/testing functions (35 y-directed, 42 z-directed, 10 γ-directed and 10 tilted y-directed) are used in the moment method. 338 magnetic current modes are used on either side of the discontinuities at z=d and z=d+s. To appreciate the modal expansion, we plot the absolute value of the y and z components of the currents existing on both sides of the substrate (see Figure 9). In this case, the microstrip line is at x=0 whereas the dipole and coplanar feed are printed at x=t. Note, as mentioned earlier, that the z-directed current is not zero at the back wall (z=0). Also, because of the microstrip geometry, the current distribution is not quite symmetric, even for the present case of θ=0° scan. Observe, via the functional trend of the current, the resonant behavior of the coplanar feedlines nearly a quarter-wavelength long. This is an indication that, particularly away from broadside, they will likely contribute to the radiation of the array. In Figures 10 and 11, the element active impedance and the polarization ratio are calculated for various thicknesses and permittivities of the substrate, and with a near-field low-permittivity dielectric radome. The unit cell dimensions as well as the microstrip and coplanar feed dimensions are identical to

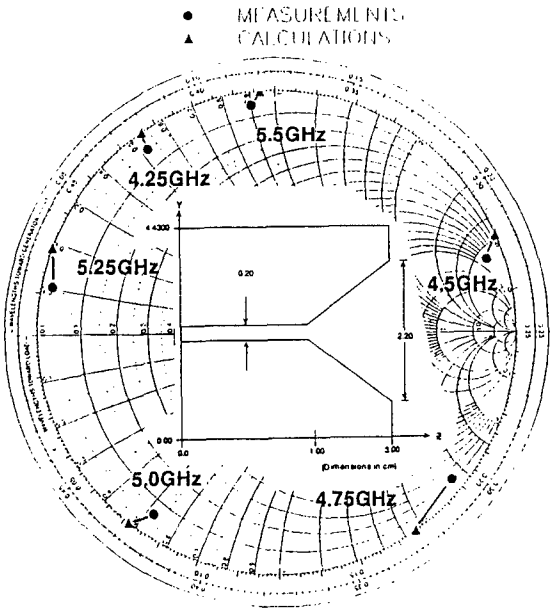


Figure 7. Theoretical and experimental ([8]) data for the reflection coefficient of an array of linearly-tapered slot antenna elements (a=4.755cm, b=4.43cm, d=10cm, ϵ_r=1).

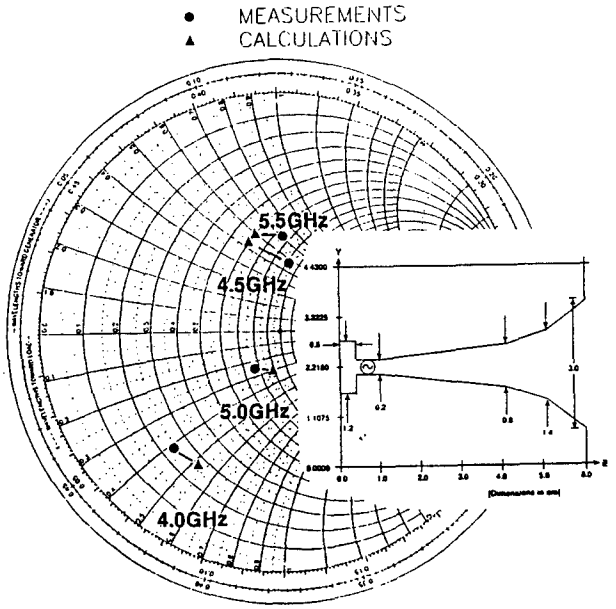


Figure 8. Theoretical and experimental ([9]) data for the reflection coefficient of an array of exponentially-tapered slot antenna elements (a=4.755cm, b=4.43cm, d=10cm, ϵ_r=1).

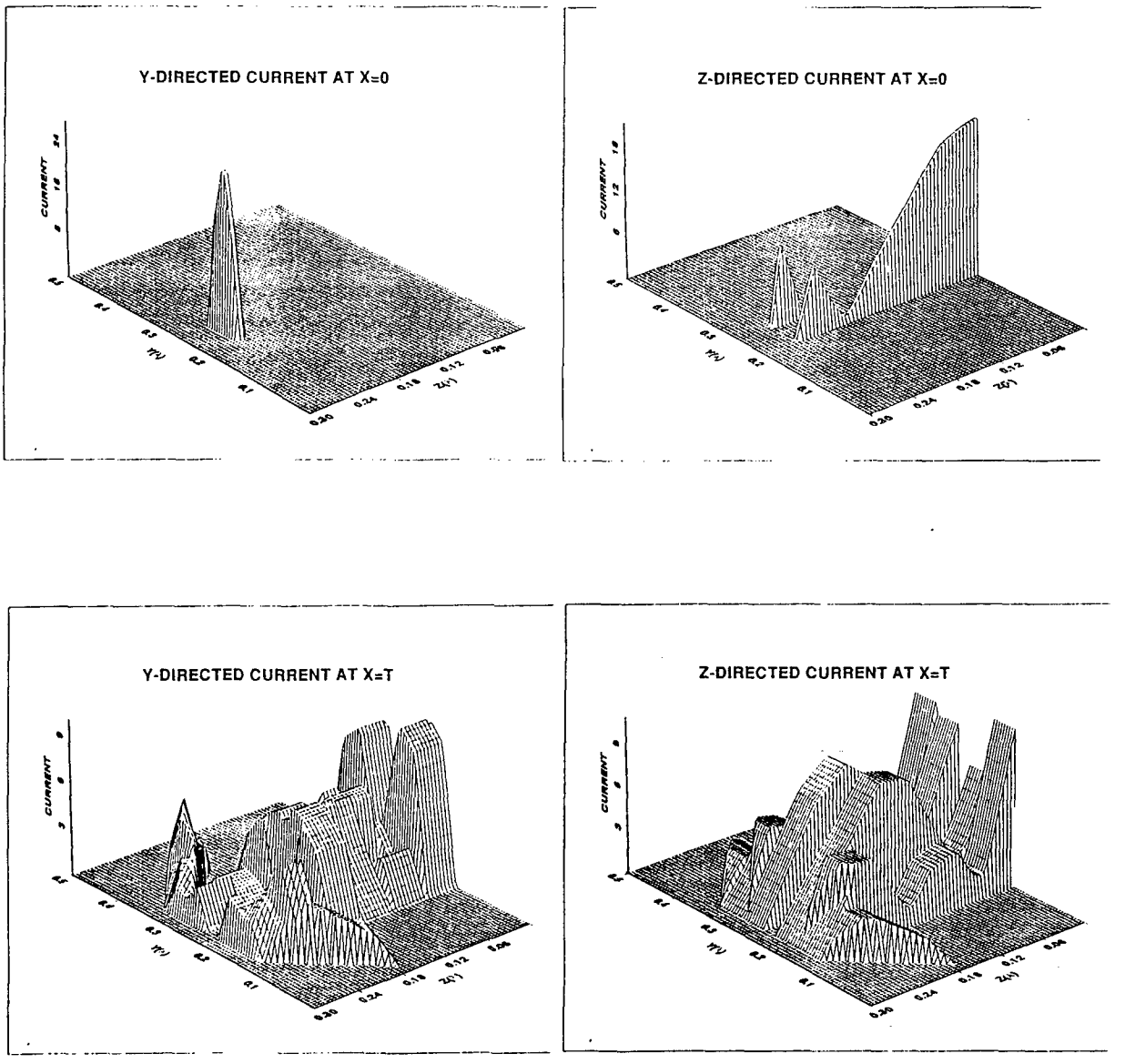
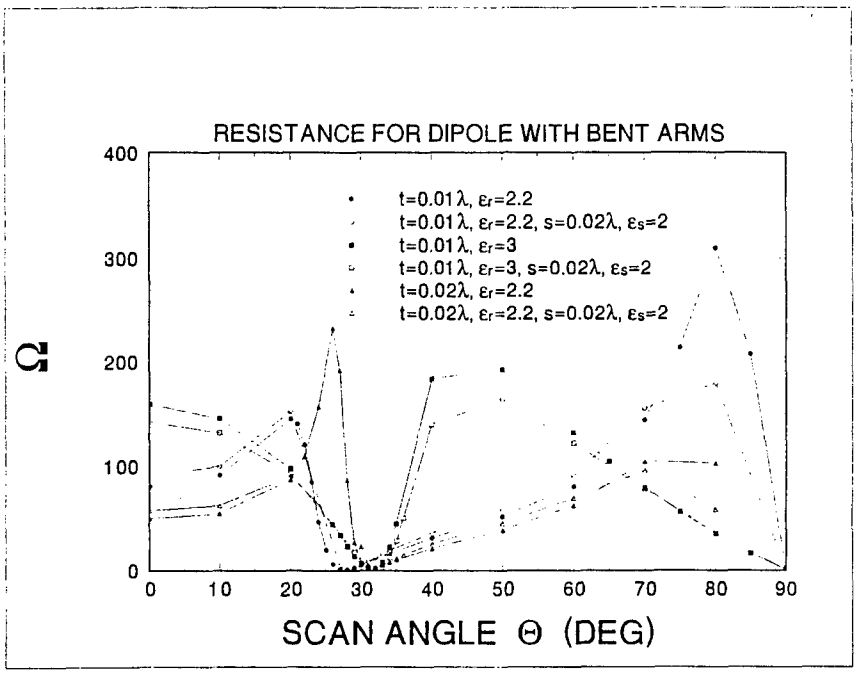


Figure 9. Three-dimensional plot of the absolute value of the y and z components of the antenna/feed current of an infinite array of bent dipoles (a=b=0.5λ, d=0.30λ, t=0.01λ, ϵ_r=2.2, θ=0°, ϕ=90°, f=300 MHz).



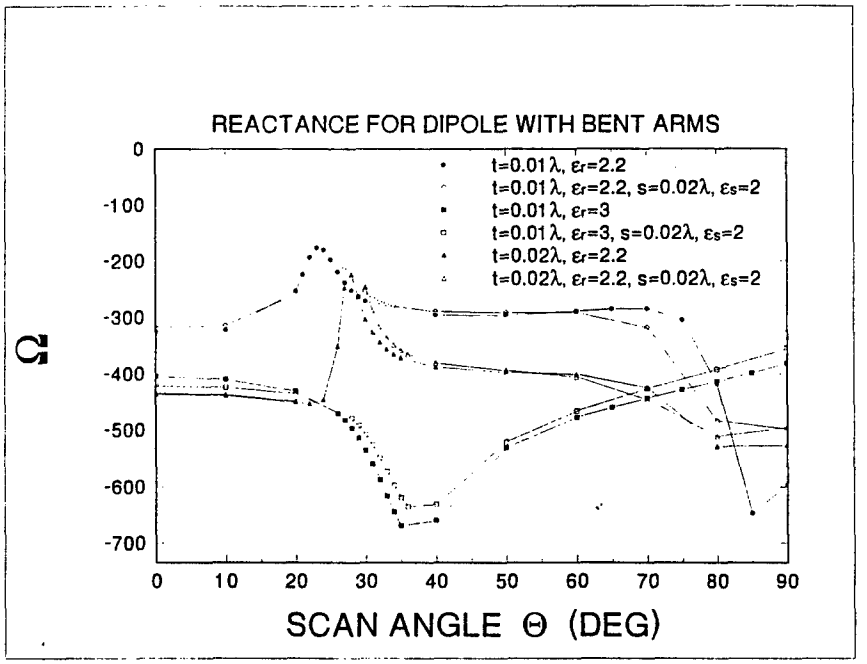


Figure 10. Active impedances versus scan for infinite arrays of bent dipoles on low-permittivity substrates (a=b=0.5λ, d=0.30λ, ϕ=90°, f=300 MHz).

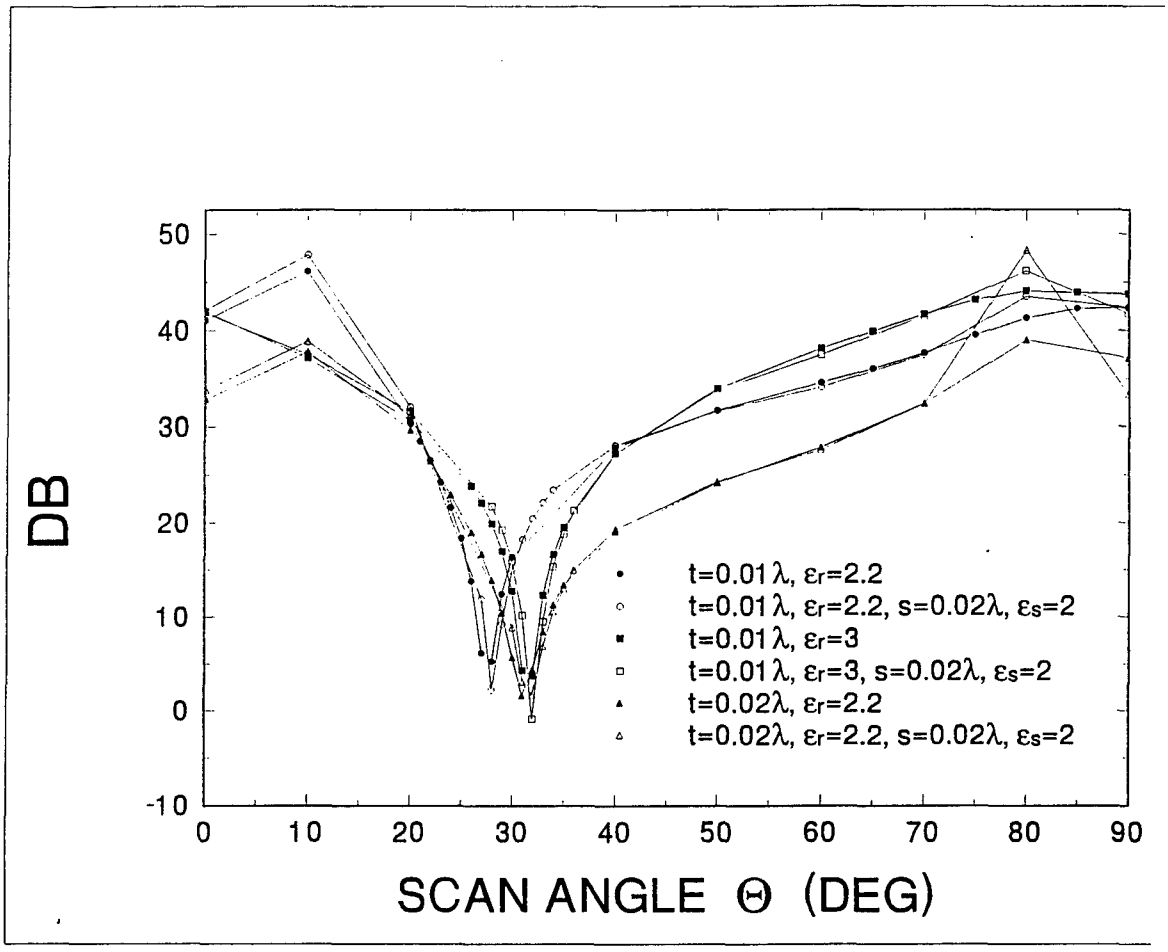
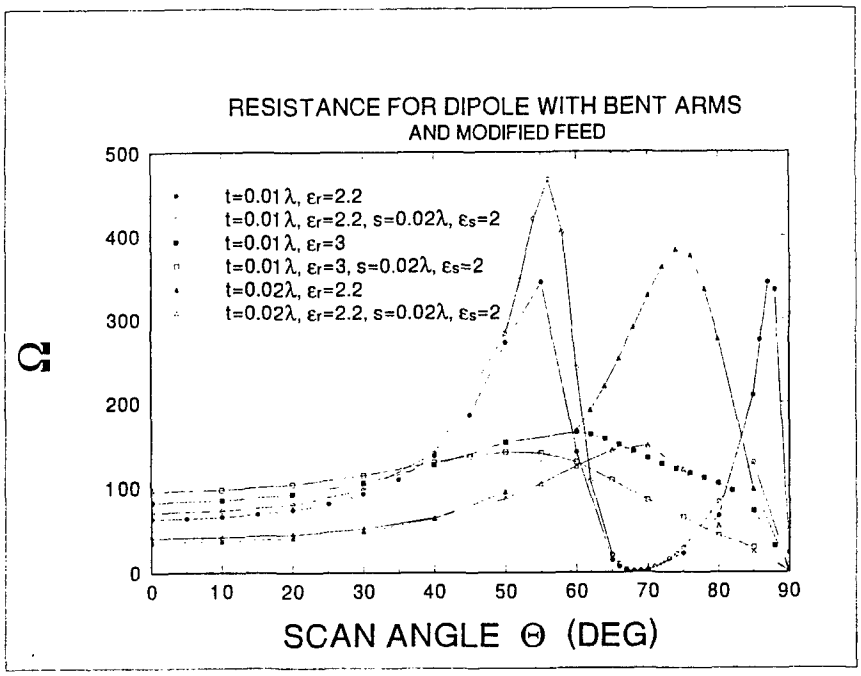


Figure 11. 20 $\mathrm{Log}_{10}(|E_\theta|/|E_\phi|)$ versus scan for infinite arrays of bent dipoles on low-permittivity substrates (a=b=0.5λ, d=0.30λ, ϕ=90°, f=300 MHz).

those of the cases shown in [4] for the straight-arm dipole element. Comparing with the results of [4], the impedance and polarization curves show a trend quite similar to that of the straight-arm dipole. In all of these cases (here and in [4]) where the array substrate has a relatively low permittivity, radiation from the feed produces a blind spot and substantial cross-polarization near 30 degree scan. There seems to be a small increase in the scan coverage by bending the arms of the dipole as the feed-induced null in the resistance curve moves a few degrees toward end-fire. As the unit cell dielectric loading increases, whether caused by a thicker or higher permittivity substrate, or by inserting a radome, the feed null also moves toward end-fire. It seems that the radiation characteristics of the dipole and coplanar feedlines are altered in such a way that, with more dielectric loading in the cell, a larger scan is required for the blind spot to appear.

Lastly, the dipole element of Figure 3 is considered. In designing such an element, the intent was to disrupt the resonant behavior of the coplanar feedlines while maintaining good ground plane characteristics for the microstrip line. Furthermore, we hoped that the resulting element could be fabricated using standard photoetching techniques. Toward that goal, small gaps are introduced in the length dimension of the coplanar stubs, forcing the z-directed currents, which are believed to be responsible for the blind spot, to be zero at various locations. The locations were selected such that the dimensions of the resulting pieces not exceed 0.18λ. In Figures 12 and 13, the active impedance and polarization ratio are plotted versus E-plane scan. As compared to Figures 10 and 11, the element of Figure 3 produces an increase of more than 20 degrees in the scan coverage of the array. For the ϵ_r=3 cases, the blind spot has disappeared from visible range offering well-behaved impedance and polarization characteristics. The proposed feed modifications seem to offer a possible avenue for increasing the scan performance of these dipole elements with low-permittivity substrates. It is important to note that while no anomaly has been detected in the H-plane of these arrays, the performance of the elements in other planes has not been studied. The reader should consider the configuration of Figure 3 as a first step toward designing wide scan dipoles, and another indication that the scan restriction seen in these dipole arrays is caused by the coplanar feedlines.



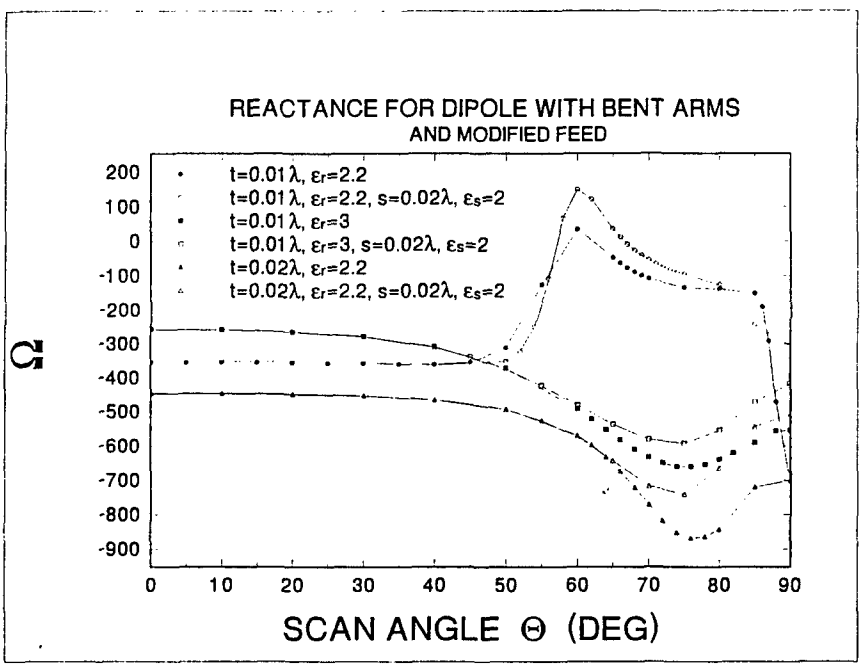


Figure 12. Active impedances versus scan for infinite arrays of bent dipoles with a modified feed (a=b=0.5λ, d=0.30λ, ϕ=90°, f=300 MHz).

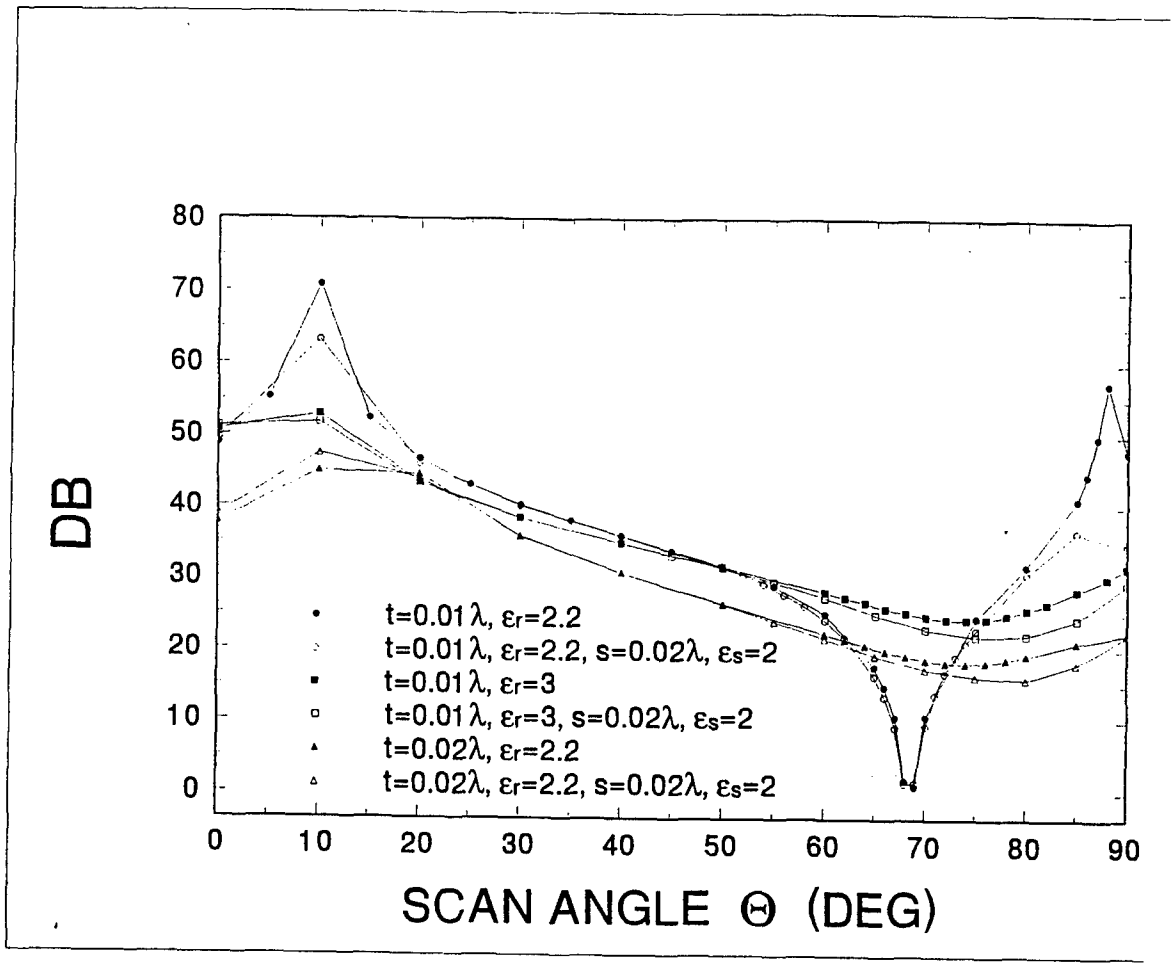


Figure 13. 20 $\mathrm{Log}_{10}(|E_\theta|/|E_\phi|)$ versus scan for infinite arrays of bent dipoles with a modified feed (a=b=0.5λ, d=0.30λ, ϕ=90°, f=300 MHz).

ACKNOWLEDGEMENT

The author wishes to thank Dr. Michael E. Cooley, Professor Daniel H. Schaubert and Dr. Boris Tomasic for their helpful suggestions. Thanks to Dr. Cooley for providing his free-space results; they were very helpful in debugging the code.

REFERENCES

[1] Bayard, J-P. R., M. E. Cooley, and D. H. Schaubert, "Analysis of Infinite Arrays of Printed Dipoles on Dielectric Sheets Perpendicular to a Ground Plane," IEEE Transactions on Antennas and Propagation, Vol. AP-39, No. 12, pp. 1722-1732, December 1991.

[2] Bayard, J-P. R., M. E. Cooley, and D. H. Schaubert, "A General Method for Treating Infinite Arrays of Antennas Printed on Protruding Dielectric Substrates," Antenna and Propagation International Symposium Digest, pp. 600-603, June 1991.

[3] Bayard, J-P. R., D. H. Schaubert, and M. E. Cooley, "E-Plane Scan Performance of Infinite Arrays of Dipoles Printed on Protruding Dielectric Substrates: Coplanar Feedline and E-Plane Metallic Wall Effects," IEEE Transactions on Antennas and Propagation, Vol. 41, No. 6, pp. 837-841, June 1993.

[4] Bayard, J-P. R., "Analysis of Infinite Arrays of Microstrip-Fed Dipoles Printed on Protruding Dielectric Substrates and Covered with a Dielectric Radome," accepted for publication in the IEEE Transactions on Antennas and Propagation.

[5] Mayer, E. D., A. Hessel, "Feed Region Modes in Dipole Phased Arrays," IEEE Transactions on Antennas and Propagation, Vol. AP-30, No. 1, pp. 66-75, January 1981.

[6] Schuman, H. K., D. R. Pflug, and L. D. Thompson, "Infinite Planar Arrays of Arbitrarily Bent Thin Wire Radiators," IEEE Transactions on Antennas and Propagation, Vol. AP-32, No. 4, pp. 364-377, April 1984.

[7] Chu, R-S., K-M. Lee, "Radiation Impedance of a Dipole Printed on Periodic Dielectric Slabs Protruding Over a Ground Plane in an Infinite Phased Array," IEEE Transactions on Antennas and Propagation, Vol. AP-35, No. 1, pp. 13-25, January 1987.

[8] Cooley, M.E., D.H. Schaubert, N.E. Buris, E.A. Urbanick, "Radiation and Scattering Analysis of Infinite Arrays of Endfire Slot Antennas With a Ground Plane," IEEE Transactions on Antennas and Propagation, Vol. AP-39, No. 11, pp. 1615-1625, November 1991.

[9] Cooley, M.E., "Analysis of Infinite Arrays of Endfire Slot Antennas," Ph.D. Dissertation, University of Massachusetts, Amherst, Massachusetts, February 1992.

COMPARING PATTERN RECOGNITION SYSTEMS

Pinyuen Chen
Associate Professor
Department of Mathematics

Syracuse University
Syracuse, NY 13244

Final Report for:
Research Initiation Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, D. C.

and

Syracuse University

December 1993

COMPARING PATTERN RECOGNITION SYSTEMS

Pinyuen Chen
Associate Professor
Department of Mathematics
Syracuse University

ABSTRACT

The main goal of this report is to study a statistical selection procedure for comparing the performances of several automatic systems which are used in neural network and artificial intelligence for the purpose of pattern recognition. Suppose that k (> 1) systems are designed to fulfill a pattern recognition purpose. The k systems are tested on images. The evaluator, based on the performance measures of the k systems that are obtained from the experiment, is interested in (1) selecting the best system if it is outstanding as compared to the remaining $k-1$ systems, or (2) selecting a subset which contains the best system and eliminates the bad systems if the best system is closed to the second best system. We propose a single-stage procedure to achieve the two goals simultaneously with certain probability requirements. The procedure parameters are given in tables for $k = 3, 4$ at the end of the report. Four examples are given in the last section to illustrate the procedure and the usages of the table.

COMPARING PATTERN RECOGNITION SYSTEMS

Pinyuen Chen

1. INTRODUCTION

This report studies a statistical procedure for comparing several pattern recognition systems (or Automatic Target Recognition (ATR) Systems in our application). A single-stage procedure is proposed to select the best system and eliminate bad systems simultaneously. The evaluator, based on the performance measures of the k (> 1) systems that are obtained from the experiment, is interested in selecting the best system if it is outstanding as compared to the remaining $k-1$ systems. If the top systems are too close for the evaluator to make a decision on which one is the best, we would eliminate bad systems and select a subset of good systems for further investigation.

Let the k competing systems be denoted by $\pi_1, \ldots, \pi_k$. All the k systems are tested on images and their performance measures are assumed to follow the same distribution with different parameter values. We will use the estimates of these unknown parameters to evaluate and compare the systems. Let μ_i $(i = 1, 2, \ldots, k)$ be the parameter associated with the system π_i. Let the ordered μ_i's be denoted by $\mu_{[1]} \leq \mu_{[2]} \leq \cdots \leq \mu_{[k]}$. The system π_k associated with $\mu_{[k]}$ is called a δ^*-best if $\mu_{[k-1]} < \mu_{[k]} - \delta^*$. A system π_i is said to be good if $\mu_i \geq \mu_{[k]} - \delta^*$, and it is said to be bad if $\mu_i < \mu_{[k]} - \delta^*$, where δ^* is a positive constant.

Though it is always desirable to choose the best system, it is reasonable to consider a system which is sufficiently close to the best, especially in the case that other factors such as the costs are also concerned in the final choice of the systems. Our formulation is a combination of the two classical approaches in statistical ranking and selection theory, namely, the indifference zone approach and the subset selection approach. This formulation was first considered by Chen and Sobel (1987) and then was applied to Automatic Target Recognition problem in Chen (1992).

This report introduces an integrated formulation to select either (1) the δ^*-best system or (2) a random-size subset containing the best and some good systems. We eliminate bad systems and at the same time, insist that the best system should be included in the selected subset. The idea of integrating the above two selecting goals is to introduce both a preference zone (PZ) and an indifference zone (IZ) (we will formally define PZ and IZ in the next section) to separate the parameter space into two disjoint sub-spaces. In the PZ, the best system is the δ^*-best, i.e., the system which is far superior to all the others. In the IZ,

we screen out all bad systems so that only the best system and those systems which are not too inferior to the best are selected. By blending these two basic formulations together, we can have added control over the selected subset size. The selection rule proposed in this report can also be used in clinical trial research where several experimental treatments are to be compared. Our procedure guarantees with certain probability that the best system is selected under any parameter configuration. This nature of the proposed procedure avoids the possibility of eliminating the best system.

In Section 2, we formally define our goal and the definitions of correct selection in the PZ and the IZ. The procedure will also be defined in Section 2. In section 3, we investigate the properties of the probability of a correct selection ($P(CS)$), the least favorable configuration (LFC) in the PZ, and the worst configuration (WC) in the IZ. Examples for choosing the procedure parameters to satisfy some preassigned probability requirements are given in Section 4 to illustrate the usages and results of our study. Tables are given at the end of the report to show the choices of different procedure parameters for various distribution parameters and probability requirements.

2. THE PROCEDURE

The observed performance measure X_i for each system π_i on an image is assumed to follow a normal distribution with unknown parameter population mean μ_i $(i = 1, 2, \ldots, k)$ with common known variance σ^2. The normal distribution assumption is reasonable since the random variable X is usually the difference of two sample means from two independent binomial experiments. (see the examples in Chen(1992) and the examples in the last section of this report) The assumption about the common variance is a more restricted one, and it needs to be generalized in future studies. Here for the purpose of data analysis, we will estimate it from the sample and consider that it has common known value σ^2 for all the k systems. Let the ordered μ_i's $(i = 1, 2, \ldots, k)$ be denoted by

$$\mu_{[1]} \leq \mu_{[2]} \leq \cdots \leq \mu_{[k]}. \tag{2.1}$$

The system that has the largest mean value $\mu_{[k]}$ is the best system. The parameter space Ω, the preference zone (PZ), and the indifference zone (IZ) are defined as follows:

$$\begin{aligned} \Omega &= \{\underset{\sim}{\mu} | \underset{\sim}{\mu} = (\mu_1, \mu_2, \ldots, \mu_k), \quad \mu_i \in R, \quad i = 1, 2, \ldots, k\} \\ PZ &= \{\underset{\sim}{\mu} | \underset{\sim}{\mu} \in \Omega, \qquad \mu_{[k]} - \mu_{[k-1]} \geq \delta^*\} \\ IZ &= \{\underset{\sim}{\mu} | \underset{\sim}{\mu} \in \Omega, \qquad \mu_{[k-1]} < \delta^*\} \end{aligned} \tag{(2.2)}$$

Our selection goal in the PZ and in the IZ are defined respectively as:

Goal in PZ: selecting only the best system. (2.3)

and

Goal in IZ: selecting a subset consisting of only good systems and containing the best system. (2.4)

Here the δ^*-best system, a good system, and the best system have already been defined in Section 1. Let CS_1 denote the selection of only the δ^*-best system and CS_2 denote the selection of a subset consisting of only the good systems and containing the best system. We need a procedure R that simultaneously satisfies the following two requirements:

$$\begin{aligned} &P(CS_1) \geq P_1^* \text{ whenever } \underset{\sim}{\mu} \in PZ \\ &\text{and } P(CS_2) \geq P_2^* \text{ whenever } \underset{\sim}{\mu} \in IZ. \end{aligned} \tag{2.5}$$

Assume that there is a common number of observations for each of the k systems. Our procedure depends on the sufficient statistics X_i for the parameter μ_i $(i = 1, 2, ..., k)$. The ordered values of the X_i's are denoted by

$$X_{[1]} \leq X_{[2]} \leq \cdots \leq X_{[k]}. \tag{2.6}$$

Let c and d be two non-negative real numbers with $c > d$. Our proposed procedure R is defined as follows:

(2.7) If $X_{[k]} - X_{[k-1]} > c$,then select the population π that gives rise to $X_{[k]}$. If $X_{[k]} - X_{[k-1]} \leq c$, then select a random-sized subset which contains all those populations π_i with $X_i > X_{[k]} - d$.

Once the procedure parameters c and d are specified, the procedure R is completely defined. For a given sample size n, the two constants c and d will be chosen to meet the probability requirements given in (2.5). For the purpose of data analysis, the evaluator usually is provided with the data and therefore he/she knows the sample size n. We will then determine c and d based on the P^* conditions and apply our procedure to make a selection. In this setting, our procedure is closer to the subset selection formulation. Examples (3) and (4) in Section A illustrate this property of our procedure. For the purpose of designing an experiment, n usually is to be determined. For a given procedure parameter c or d, we can solve n and the other procedure parameter by the two simultaneous inequalities given in (2.6). This setting is closer to the indifference zone formulation. Examples (1) and (2) in Section A illustrate this property of our procedure.

3. THE PROBABILITY OF A CORRECT SELECTION AND ITS PROPERTIES

Let $X_{(r)}$ be the observed sample mean from the system associated with mean $\mu_{[r]}$. In the PZ, a correct selection CS_1 occurs if and only if we select only the δ^*-best system associated with mean $\mu_{[k]}$. We first prove the following theorem concerning the least favorable configuration (LFC), the parameter configuration in the PZ that gives the smallest probability of a correct selection. We also assume that each X_i is a normal random variable with mean μ_i and variance σ^2.

Theorem 3.1 Under procedure R, the LFC for $P(CS_1)$ in the PZ has the following form:

$$\mu_{[1]} = \mu_{[2]} = \cdots = \mu_{[k]} - \delta^* \tag{3.1}$$

Proof. Consider an arbitrary $\mu \in PZ$. From the definition of our procedure in (2.8), we can write $P(CS_1)$ as

$$\begin{aligned} P(CS_1|PZ) &= P(X_{(k)} > X_{(i)} + c, \qquad i = 1, 2, \ldots, k-1) \\ &= P\left(\frac{X_{(i)} - \mu_{[i]}}{\sigma} < \frac{X_{(k)} - \mu_{[k]}}{\sigma} - \frac{c}{\sigma} + \frac{\mu_{[k]} - \mu_{[i]}}{\sigma}, \qquad i = 1, 2, \ldots, k-1\right) \\ &= \int_{-\infty}^{\infty} \prod_{i=1}^{k-1} \Phi\left(x - \frac{c}{\sigma} + \frac{\mu_{[k]} - \mu_{[i]}}{\sigma}\right) d\Phi(x) \\ &\leq \int_{-\infty}^{\infty} \Phi^{k-1}\left(x - \frac{c}{\sigma} + \frac{\delta^*}{\sigma}\right) d\Phi(x). \end{aligned} \tag{3.2}$$

The last inequality follows from the fact that, in the PZ, $\mu_{[k]} - \mu_{[i]} \geq \delta^*$. This complete the proof of the theorem.

Now we deal with the so-called worst configuration (the parameter configuration in the IZ that minimizes the probability of a correct selection.) Let Ω_t denote the subset in Ω whose elements have exactly t good populations. That is,

$$\Omega_t = \{\underset{\sim}{\mu} | \mu_{[j]} > \mu_{[k]} - \delta^*, \quad j = k-t+1, \ldots, k; \quad \mu_{[i]} \leq \mu_{[k]} - \delta^*, \quad i = 1, 2, \ldots, k-t\}. \tag{3.3}$$

It is clear that Ω_t, $t = 2, 3, \ldots, k$ form a partition of the IZ. The probability of a correct selection in each of the Ω_t can be expressed as

$$\begin{aligned} &P(CS_2|\Omega_t) \\ &= P(X_{(k-t+1)}, \ldots, X_{(k-1)} \leq X_{(k)}, X_{(1)}, \ldots, X_{(k-t)} \leq X_{(k)} - d) \\ &+ P(X_{(1)}, \ldots, X_{(k-t)} \leq X_{(k-1)} - d < X_{(k)} \leq X_{(k-1)}, X_{(k-t+1)}, \ldots, X_{(k-2)} \leq X_{(k-1)}) \\ &+ \ldots \\ &+ P(X_{(s)}, \ldots, X_{(k-t)} \leq X_{(k-t+1)} - d < X_{(k)} < X_{(k-t+1)}, X_{(k-t+2)}, \ldots, X_{(k-1)} < X_{(k-t+1)}) \end{aligned} \tag{3.4}$$

For the case that $t = k$, we have from (3.4) that

$$
\begin{aligned}
& P(CS_2|\Omega_k) \\
& \quad = P(X_{(1)}, \ldots, X_{(k-1)} \le X_{(k)}) \\
& \quad + P(X_{(1)}, \ldots, X_{(k-2)} < X_{(k-1)}, X_{(k-1)} - d < X_{(k)} < X_{(k-1)}) \\
& \quad + \ldots \\
& \quad + P(X_{(2)}, \ldots, X_{(k-1)} < X_{(1)}, X_{(1)} - d < X_{(k)} < X_{(1)}).
\end{aligned} \tag{3.5}
$$

Define $M_0 = \max\{y_\alpha, \alpha = 1, 2, \ldots, k\}$ and $\psi = \psi(y_1, y_2, \ldots, y_k)$ by

$$
\psi = \begin{cases} 1, & \text{if } y_k \ge y_i,\ i = 1, 2, \ldots, k-1; \\ 1, & \text{if } M_0 - d < y_k \le M_0 = y_i,\ i = 1, 2, \ldots, k-1; \\ 0, & \text{otherwise.} \end{cases} \tag{3.6}
$$

It is clear that ψ is a non-decreasing function in y_k while we hold all $y_i, i = 1, 2, \ldots, k-1$ fixed, and it is a non-increasing function in y_i, $i \neq k$ while we hold all other y_j, $j = 1, 2, \ldots, k-1$, fixed. Replacing y_i in (3.6) by $X_{(i)}$, the first two cases in (3.6) cover exactly the event of a correct selection CS_2 in Ω_k. It follows from a lemma in Alam and Rizvi (1966) (see Appendix) that, for $\underset{\sim}{\mu} \in \Omega_k$, $P(CS_2|\Omega_k)$ is a non-decreasing function in $\mu_{[k]}$ and a non-increasing function in $\mu_{[i]}$ for $i = 1, 2, \ldots, k-1$. We have just proved the following theorem.

Theorem 3.2 Under procedure R, the worst configuration WC_k of $P(CS_2)$ in Ω_k is of the form:

$$
\mu_{[1]} = \mu_{[2]} = \cdots = \mu_{[k]} \tag{3.7}
$$

Now we can write

$$
\begin{aligned}
& P(CS_2|\Omega_k) \\
& \quad = \int_{-\infty}^{\infty} \prod_{i=1}^{k-1} \Phi(x + \tfrac{\mu_{[k]} - \mu_{[i]}}{\sigma}) d\Phi(X) \\
& \quad + \int_{-\infty}^{\infty} \prod_{i=1}^{k-2} \Phi(x + \tfrac{\mu_{[k-1]} - \mu_{[i]}}{\sigma}) \left[\Phi(x - \tfrac{\mu_{[k]} - \mu_{[k-1]}}{\sigma}) - \Phi(x - \tfrac{\mu_{[k]} - \mu_{[k-1]}}{\sigma}) - \frac{d}{\sigma} \right] d\Phi(x) \\
& \quad + \ldots \\
& \quad + \int_{-\infty}^{\infty} \prod_{i=2}^{k-1} \Phi(x + \tfrac{\mu_{[1]} - \mu_{[i]}}{\sigma}) \left[\Phi(x - \tfrac{\mu_{[k]} - \mu_{[1]}}{\sigma}) - \Phi(x - \tfrac{\mu_{[k]} - \mu_{[1]}}{\sigma} - \frac{d}{\sigma}) \right] d\Phi(x) \\
& \quad \ge \int_{-\infty}^{\infty} \Phi^{k-2}(x) \left[k\Phi(x) - (k-1)\Phi(x - \frac{d}{\sigma}) \right] d\Phi(x)
\end{aligned} \tag{3.8}
$$

In general, for $t = 2, \ldots, k-1$, using the same argument as above and the lemma of Alam and Rizvi we can show that $P(CS_2|\Omega_k)$ in (3.4) is a non-decreasing function in $\mu_{[k]}$ while we hold all the $\mu_{[i]}$, $i =$

$1, 2, \ldots, k-1$ fixed and it is a non-increasing function of $\mu_{[i]}$ for $i = 1, 2, \ldots, k-t$ while we hold all other $\mu_{[j]}, j \neq i$ fixed. Therefore, the infimum of $P(CS_2|\Omega_x)$ must occur when

$$\mu_{[1]} = \mu_{[2]} = \cdots = \mu_{[k-t]} = \mu_{[k]} - \delta^*. \tag{3.9}$$

Although we are not able to use the same argument as above to push the values of $\mu_{[k-t+1]}, \mu_{[k-2+2]}, \cdots, \mu_{[k-1]}$ to the other end $\mu_{[k]}$, numerical plottings of the function in (3.4) show that the $P(CS_2|\Omega_t)$ is always a unimodal function of $\delta_i = \mu_{[k]} - \mu_{[k-i]}$ for $i = 1, 2, \ldots, t-1$, and $0 \leq \delta_i \leq \delta^*$. Thus the lowest value of $P(CS|\Omega_t)$ always occurs at $\delta_i = 0$ or $\delta_i = \delta^*$. The worst configuration is therefore conjectured at either (3.1) or at

$$\mu_{[1]} = \mu_{[2]} = \cdots = \mu_{[k-t]} = \mu_{[k-t+1]} - \delta^* = \cdots = \mu_{[k]} - \delta^*. \tag{3.10}$$

For a given k, the worst configuration WC over the entire IZ can therefore be obtained by

$$WC = \left\{ \underset{\sim}{\mu} \in \{WC_t, \quad t = 2, \ldots, k\} | P(CS_2|\underset{\sim}{\mu}) = \min_{2 \leq t \leq k} \{P(CS_2|WC_t)\} \right\}. \tag{3.11}$$

4. SPECIAL CASES AND EXAMPLES

We first consider several special cases of $P(CS_2)$.

(CASE 1) $k = 2$:

From Theorem 3.1 and 3.2, we have

$$P(CS_2|PZ) \geq P(CS_2|LFC) = \int_{-\infty}^{\infty} \Phi(X - \frac{c}{\sigma} + \frac{\delta^*}{\sigma}) d\Phi(X) \tag{4.1}$$

From (3.8) we have

$$P(CS_2|IZ) \geq \int_{-\infty}^{\infty} \left(2\Phi(X) - \Phi(X - \frac{d}{\sigma})\right) d\Phi(X) \tag{4.2}$$

Set the above two integrals (the right hand sides of the two inequalities) at P_1^* and P_2^*, respectively. Then the procedure parameters c and d can be solved from these two simultaneous equations.

(CASE 2) $k = 3$:

Take $\Delta = \frac{\delta^*}{\sigma/\sqrt{n}} = 5.00$. From (3.1) and (3.2), we have

$$P(CS_1|PZ) = P(X_{(3)} > X_{(i)} + c, \quad i = 1, 2) \geq \int_{-\infty}^{\infty} \Phi^2(X - \frac{c}{\sigma} + \frac{\delta^*}{\sigma}) d\Phi(X) \tag{4.3}$$

Set the above integral at $P_1^* = .90$, we obtain $C = \frac{c}{\sigma/\sqrt{n}} = 2.770$. Now we can solve d from $P(CS_2|WC_2)$ and $P(CS_2|WC_3)$ under the restriction that $D < C = 2.770$, and

$$\min\{P(CS_2|WC_2), \quad P(CS_2|WC_3)\} \geq P_2^* = .90. \tag{4.4}$$

When $t = 2$, we have

$$\begin{aligned} & P(CS_2|\Omega_2) \\ & = P(X_{(2)} < X_{(3)}, \quad X_{(1)} < X_{(3)} - d) + P(X_{(2)} - d < X_{(3)} \leq X_{(2)}, \quad X_{(1)} < X_{(2)} - d) \\ & \geq \int_{-\infty}^{\infty} \{\Phi(X - D + \Delta)\Phi(X + T) + \Phi(X - D + \Delta - T)[\Phi(X - T) - \Phi(X - T - D)]\} d\Phi(X) \end{aligned} \tag{4.5}$$

Here $D = \frac{d}{\sigma}$, $\Delta = \frac{\delta^*}{\sigma}$, and $T - \frac{\mu_{(3)} - \mu_{[2]}}{\sigma}$. The last inequality in (4.5) is from (3.9). It is clear from the numerical plotting of Table 1 at the end of the report that, when $D < 2.50$, WC_2 is at $T = 0$, and at $T = \Delta$ otherwise. Let $\Delta = 5.0$, we have

$$\begin{aligned} P(CS_2)|\Omega_2)|_{T=0} &= \int_{-\infty}^{\infty} \Phi(X - D + 5) \left[2\Phi(X) - \Phi(X - D)\right] d\Phi(X) \\ P(CS_2|\Omega_2)|_{T=\Delta} &= \int_{-\infty}^{\infty} \Phi(X + 5)\Phi(X - D + 5) + \Phi(X - D) \left[\Phi(X - 5) - \Phi(X - D - 5)\right] d\Phi(X) \\ P(CS_2|WC_3) &= \int_{-\infty}^{\infty} \Phi(X) \left[3\Phi(X) - 2\Phi(X - D)\right] d\Phi(X). \end{aligned} \tag{4.6}$$

If we set $P_2^* = .90$, $D = 2.25$, then $P(CS_2|WC_3) \geq .9026$, $P(CS_2|\Omega_2) \geq .9394$. Combine the above inequalities with $P(CS_1|PZ) \geq P_1^* = .90$, we conclude that, by choosing $C = 2.770$ and $D = 2.250$, our procedure guarantees that $P_1^* = P_2^* = .90$.

(CASE 3) $k = 4$:

The first P^* condition is established by solving for c value in (3.2). As for the solution of d, we again need to find the lower bound of $P(CS_2|IZ)$ among all cases of $P(CS_2|WC_t)$ for $t = 2, 3, 4$, and then solve for d for the preassigned P_2^* value. The WC_4 here is just the EPC; The WC_3, similar to the case of $k = 3$ and $t = 2$, is either at (3.1) or (3.7). The only case needs to be taken care here is when $t = 3$, i. e., the case when there are 3 good systems among 4 possible systems. The probability of a correct selection can be written as

$$
\begin{aligned}
& P(CS_2|\Omega_3) \\
& = P(X_{(2)}, X_{(3)} \leq X_{(4)}, X_{(1)} < X_{(4)} - d) \\
& + P(X_{(1)} \leq X_{(3)} - d < X_{(4)} \leq X_{(3)}, X_{(2)} \leq X_{(3)}) \\
& + P(X_{(1)} \leq X_{(2)} - d < X_{(4)} \leq X_{(2)}, X_{(3)} \leq X_{(2)}) \\
& \geq \int_{-\infty}^{\infty} \Phi(X + T_2)\Phi(X + T_3)\Phi(X - D - \Delta) d\Phi(X) \\
& + \int_{-\infty}^{\infty} \Phi(X - D + \Delta - T_3)\Phi(X + T_2 - T_3)[\Phi(X - T_3)] - \Phi(X - T_3 - D)] d\Phi(X) \\
& + \int_{-\infty}^{\infty} \Phi(X - D + \Delta - T_2)\Phi(X - T_2 + T_3)[\Phi(X - T_2) - \Phi(X - T_2 - D)] d\Phi(X)
\end{aligned}
$$

Here $T_2 = \frac{\mu_{[4]} - \mu_{[2]}}{\sigma}$, $T_3 = \frac{\mu_{[4]} - \mu_{[3]}}{\sigma}$. Numerical plotting shows that the sum of the integrals on the right- hand side of (4.7) is a unimodal function of T_2 when T_3 is held fixed. Thus the minimum of $P(CS_2|\Omega_3)$ always occurs at either $T_2 = T_3$ or $T_2 = \Delta$.

Table 2 at the end of the report gives the procedure parameters C and D for given k, Δ, and p^* values. IMSL routines (version 10.0) were used to evaluate the integrations involved in the equations. It should be noted that the entries of C and D depend on the choices of Δ values. For example, if $k = 4$, $P_1^* = .90$, $P_2^* = .75$, and $\Delta = 5.00$, then from Table 1 we obtain $C = 2.548$, $D = 1.680$. However if we take $\Delta = 7.00$ instead, the answers should be $C = 4.548$, $D = 1.680$. From our numerical plotting, for the case $k = 3$, and 4, the overall minimum of $P(CS_2|IZ)$ seems always to occur at the worst configuration of $t = k$, that is, at WC_k. Hence, as shown in Table 2, the choice of the procedure parameter D is independent of the Δ value. We conjecture that this property is true for any k.

The performance of the proposed procedure may be evaluated by the expected selected subset size and the total sample size for certain P^*-conditions. We hope to conduct further study on this effort.

In the following, we will consider four examples to illustrate the usage of Table 2.

EXAMPLE 1. If an evaluator wishes to select the best system among 3 competing systems with confidence level .90 whenever the distance between the true performance measures of the best and the second best is at least .05, otherwise to select a subset that contains the best systems and consists of only good systems with confidence level also at .90. Based on the past experience, the evaluator requires that the competing systems produce at least .90 for $P(D)$, the probability of a detection, and at most .01 for $P(FA)$, the probability of a false alarm. (See for example, Figure 3.3 NNIIES Operating Curve in (6)). Then a conservative upper bound for σ^2 can be estimated as follows:

$$\sigma^2 = V_m(V) + V_m(W) \leq (.90)(.10)/n + (.01)(.99)/m = .09/n + .0099/m < .0999/n$$

where n is the number of targets and m is the total number of false alarm opportunities and $n < m$.

From Table 2, the procedure parameter (C, D) exists when Δ is fixed at 5. It is also clear from (4.3) and (4.5) respectively that both $P(CS_1|PZ)$ and $P(CS_2|IZ)$ are increasing in Δ. Thus as long as $n > (5)(5)(.0999)/((.05)(.05)) = 900$, we have $\Delta = \dfrac{\delta^*}{\sigma} > 5$. Hence we can guarantee that our procedure with procedure parameter $(C, D) = (2.770, 2.200)$ will achieve our goal.

EXAMPLE 2. If the evaluator in EXAMPLE 1 requires higher probability requirements $P_1^* = .95$, and $P_2^* = .95$, then from the same table as above, we are not able to find procedure parameter to accomplish our goal. However, if we increase the Δ value to 6.0, we find the solution for (C, D) at $(3.290, 2.600)$. Here $\Delta = 6.0$ means a larger sample size. The sample size n required here is at least $(6)(6)(.0999)/((.05)(.05)) = 1438.56$.

EXAMPLE 3. Suppose that 4 competing systems π_1, π_2, π_3, π_4 are tested with the same images used in evaluating NNIIES system (see (4)) and they give the following $P(FA)$'s and $P(D)$'s:

	System π_1	System π_2	System π_3	System π_4
$P(D)$	230/285	247/285	234/285	212/285
$P(FA)$	81/6499	102/6499	181/6499	188/6499

The evaluator wants to make sure, with probability at least .90, that he selects the best system if the best performance measure is at least $\delta^* = .1$ higher than the second best, or he will select a subset containing the best and eliminating all the bad systems, with a probability of at least .90. Thus we have $k = 4$,

$P_1^* = .90$, $P_2^* = .90$ and $\delta^* = .1$. Suppose that from the past experience, the common sample variance can be estimated at .0004 (see example 1 in (2)). Thus $\Delta = \delta^*/\sigma = .1/.02 = 5$. From Table 2, we can find that the (C, D) that guarantees the requirements is $(2.548, 2.450)$. Therefore our procedure parameters are $c = 2.548 \times .02 = .05096$ and $d = 2.45 \times .02 = .0490$. The sample performance measures of the four systems are .7946, .8510, .7932, and .7149 respectively. Since $.8510 - .7946 = .0564 > c$, we select the system π_2 as the best system with confidence level .90.

EXAMPLE 4. Suppose that the same data values in EXAMPLE 3 are used again, but with a new sample variance $\sigma^2 = .000625$. Thus $\Delta = 4.0$. From our Table 2, we can see that there does not exist procedure parameter (c, d) that satisfies our probability requirements $P_1^* = P_2^* = .90$. Suppose that we relax our requirements to $P_1^* = .75$, $P_2^* = .75$. Then from the same Table 2, we find $(C, D) = (2.317, 1.680)$. Thus $c = C \times \sigma = 2.317 \times .025 = .057925$, and $d = D \times \sigma = 1.680 \times .025 = .042$. Since $.8510 - .7946 = .0564 < c$, we select all the systems with performance measures at least $.8510 - .042 = .809$. Thus we include only the system π_2 in the selected subset and claim with confidence level .75 that the systems π_1, π_3, and π_4 are all bad systems.

Table 1

Numerical Results for $P(CS_2|\Omega_2)$
with $k = 3$, $t = 2$, and $\Delta = 5.00$

$P(CS_2|\Omega_2)$

T/D	0.00	0.25	0.50	0.75	1.00	1.25	1.50
0.0	.5000	.5701	.6381	.7020	.7601	.8113	.8549
0.5	.6382	.7020	.7602	.8115	.8552	.8914	.9200
1.0	.7602	.8116	.8555	.8918	.9208	.9430	.9592
1.5	.8555	.8919	.9211	.9437	.9605	.9723	.9798
2.0	.9213	.9440	.9611	.9734	.9817	,9868	.9890
2.5	.9614	.9739	.9826	.9883	.9917	.9930	.9924
3.0	.9829	.9890	.9928	.9949	.9957	.9952	.9934
3.5	.9932	.9957	.9970	.9975	.9971	.9959	.9935
4.0	.9975	.9983	.9986	.9983	.9975	.9960	.9934
4.5	.9991	.9992	.9991	.9986	.9976	.9960	.9934
5.0	.9996	.9995	.9992	.9987	.9977	.9960	.9934

T/D	1.75	2.00	2.25	2.50	2.75	3.00	3.25
0.0	.8906	.9187	.9394	.9530	.9599	.9600	.9531
0.5	.9416	.9568	.9660	.9694	.9671	.9586	.9433
1.0	.9700	.9759	.9772	.9737	.9650	.9505	.9292
1.5	.9835	.9835	.9799	.9720	.9594	.9409	.9156
2.0	.9886	.9853	.9789	.9686	.9535	.9327	.9051
2.5	.9899	.9850	.9771	.9655	.9491	.9271	.8984
3.0	.9899	.9842	.9757	.9634	.9465	.9239	.8947
3.5	.9896	.9836	.9748	.9623	.9451	.9223	.8930
4.0	.9894	.9833	.9743	.9617	.9445	.9217	.8923
4.5	.9893	.9831	.9742	.9615	.9443	.0214	.8921
5.0	.9892	.9831	.9741	.9615	.9442	.9214	.8921

T/D	3.50	3.75	4.00	4.25	4.50	4.75	5.00
0.0	.9387	.9160	.8845	.8435	.7930	.7336	.6665
0.5	.9205	.8892	.8489	.7994	.7411	.6750	.6031
1.0	.9003	.8629	.8167	.7618	.6990	.6299	.5566
1.5	.8827	.6415	.7920	.7344	.6699	.6002	.5273
2.0	.8700	.8270	.7759	.7147	.6527	.5833	.5115
2.5	.8623	.8158	.7670	.7084	.6439	.5752	.5042
3.0	.8583	.8143	.7628	.7043	.6402	.5718	.5013
3.5	.8565	.8125	.7611	.7028	.6388	.5706	.5004
4.0	.8559	.8119	.7605	.7022	.6383	.5703	.5001
4.5	.8557	.8117	.7603	.7021	.6382	.5702	.5000
5.0	.8556	.8116	.7603	.7021	.6382	.5702	.5000

Note: underline '_____' in each column indicates the occurence of the minimum $P(CS_2|\Omega_2)$ for the D value,
$\Delta = \frac{\delta^*}{\sigma/\sqrt{n}}$, $D = \frac{d}{\sigma/\sqrt{n}}$, and $T = \frac{\mu_{(3)} - \mu_{(2)}}{\sigma/\sqrt{n}}$.

Table 2

The procedure parameters (C, D) for
given k, Δ, P_1^*, and P_2^*

k=3

$\Delta = 1.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	*	*	*
.90	*	*	*
.95	*	*	*

$\Delta = 2.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(0.565, *)	*	*
.90	(0.565, *)	*	*
.95	(0.565, *)	*	*

$\Delta = 3.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(1.565, 1.400)	(0.770, *)	(0.290, *)
.90	(1.565, *)	(0.770, *)	(0.290, *)
.95	(1.565, *)	(0.770, *)	(0.290, *)

$\Delta = 4.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(2.565, 1.400)	(1.770, 1.400)	(1.290. *)
.90	(2.565, 2.200)	(1.770, *)	(1.290, *)
.95	(2.565, 1.400)	(1.770, *)	(1.290, *)

$\Delta = 5.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(3.565, 1.400)	(2.770, 1.400)	(2.290, 1.400)
.90	(3.565, 2.200)	(2.770, 2.200)	(2.290, 2.200)
.95	(3.565, 2.600)	(2.770, 2.600)	(2.290, *)

$\Delta = 6.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(4.565, 1.400)	(3.770, 1.400)	(3.290, 1.400)
.90	(4.565, 2.200)	(3.770, 2.200)	(3.290, 2.200)
.95	(4.565, 2.600)	(3.770, 2.600)	(3.290, 2.600)

Note: 1. For $\Delta = 6.0 + i$the formulas for C, D are:
$C = 4.565 + i.3.770 + i.3.290 + i$ for $P_1^* = .75, .90, .95$ respectively;
$D = 1.400, \quad 2.200, \quad 2.600$ for $P_2^* = .75, .90, .95$ respectively.
2. * indicates that no solution exists under the D value and the probability requirements.

Table 2 (continued)

The procedure parameters (C, D) for
given k, Δ, P_1^*, and P_2^*

k=4

$\Delta = 1.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	*	*	*
.90	*	*	*
.95	*	*	*

$\Delta = 2.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(0.317, *)	*	*
.90	(0.317, *)	*	*
.95	(0.317, *)	*	*

$\Delta = 3.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(1.317, *)	(0.548, *)	(0.083, *)
.90	(1.317, *)	(0.548, *)	(0.083, *)
.95	(1.317, *)	(0.548, *)	(0.083, *)

$\Delta = 4.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(2.317, 1.680)	(1.548, *)	(1.083. *)
.90	(2.317, *)	(1.548, *)	(1.083, *)
.95	(2.317, 1.680)	(1.548, *)	(1.083, *)

$\Delta = 5.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(3.317, 1.680)	(2.548, 1.680)	(2.083, 1.680)
.90	(3.317, 2.450)	(2.548, 2.450)	(2.083, *)
.95	(3.317, 2.915)	(2.548, *)	(2.083, *)

$\Delta = 6.0$

$P_2^* \backslash P_1^*$	.75	.90	.95
.75	(4.317, 1.680)	(3.548, 1.680)	(3.083, 1.680)
.90	(4.317, 2.450)	(3.548, 2.450)	(3.083, 2.450)
.95	(4.317, 2.915)	(3.548, 2.915)	(3.083, 2.915)

Note: 1. For $\Delta = 6.0 + i$the formulas for C, D are:
$C = 4.565 + i.3.770 + i.3.290 + i$ for $P_1^* = .75, .90, .95$ respectively;
$D = 1.400, \quad 2.200, \quad 2.600$ for $P_2^* = .75, .90, .95$ respectively.
2. * indicates that no solution exists under the D value and the probability requirements.

APPENDIX

Lemma (due to Alam and Rizve (1966)): Let $\underset{\sim}{X} = (X_1, X_2, \ldots, X_k)$ be a vector-valued random variables of $k > 0$ independent components such that for each i, the random variable X_i has the distribution function $H(X_i, \theta_i)$, which is non-increasing in θ_i for constant X_i, $i = 1, \ldots, k$. If $\Psi(\underset{\sim}{X})$ is monotone function of X_i when the other components are fixed then $E\Psi(\underset{\sim}{X})$ is monotone in θ_i in the same direction.

REFERENCES

(1) Alam, K. and Rizvi, M. H. (1966) *Selection from multivariate normal populations,* Ann. Inst. Statist. Math., **18,** 307-318.

(2) Chen, P. (1992) *Statistical comparison of several automatic target recognition (ATR) systems,* Final Report, 1992 AFOSR Summer Faculty Research Program.

(3) Chen, P. and Sobel, M. (1987) *An integrated formulation for selecting the t best of k normal populations,* Communications of Statistics, Theory & Methods, **16**(1), 121-146.

(4) Weingard, F., Thoet, B., Slutz, L., Fossum, J., and Rainey, T. (1992) *Neural Network Investigation,* Technical Report RL-TR-92- 10, Rome Laboratory, Air Force Systems Command, Griffiss Air Force Base, NY 13441-5700.

WIDEBAND ATM NETWORKS FOR THE DYNAMIC THEATER ENVIRONMENT

Robert R. Henry
Professor
Department of Electrical & Computer Engineering

University of Southwestern Louisiana
P.O. Box 43890
Lafayette, LA 70504-3890

Final Report for:
Research Extension Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, D.C.

and

The University of Southwestern Louisiana

May 24, 1993

WIDEBAND ATM NETWORKS FOR THE DYNAMIC THEATER ENVIRONMENT

Robert R. Henry
Professor
Department of Electrical & Computer Engineering

ABSTRACT

An underlying concept of the global network is that the National Command Authority (NCA) will be able to communicate directly with personnel in the tactical environment. In order that graphics and video information be delivered in a timely manner, high capacity links are essential. ATM is a candidate for the wide-area part of this network since it is wideband and is being developed commercially. However, since ATM is designed for peacetime (well-behaved) use, STIP-like protocols must be incorporated to provide for survivability in the tactical area. This report presents research results that formulates methods which combine the advantages of both the ATM and STIP protocols into a network suitable for the military environment.

WIDEBAND ATM NETWORKS FOR THE DYNAMIC THEATER ENVIRONMENT

Robert R. Henry

INTRODUCTION

Two of the significant projects currently underway in the Telecommunications Division of Rome Laboratory deal with military communications networks. The Evaluation and Development of Multimedia Networks Under Dynamic Stress (EDMUNDS) project [1] is an ongoing effort to develop Secure Tactical Internet Protocols (STIP) that respond well to a stressed tactical environment. Such an environment is characterized by communication links subject to jamming and nodes subject to destruction by the enemy. The assumption is that the links have relatively low bandwidth, and that there is sufficient time to perform sophisticated processing to determine optimal datagram routing.

The Secure Survivable Communication Network (SSCN) project [2] was initiated in an effort to utilize the emerging Broadband Integrated Services Digital Network (B-ISDN) for military applications. This network relies on the Asynchronous Transfer Mode (ATM) [3] to provide rapid multiplexing and routing of data over wide bandwidth links. Due to high data rates there is relatively little time to determine the optimal route of each packet (cell). Thus connection oriented services have been selected to provide routing that is fixed for the call duration. This is in direct contrast with the STIP routing protocol.

A WIDEBAND DYNAMIC NETWORK

Each of the networks discussed above may be classified into one of two groups based on link capacity and the ability to adapt to a stressed environment. The classification is as follows:

ATM: high capacity, low adaptability

STIP: low capacity, high adaptability.

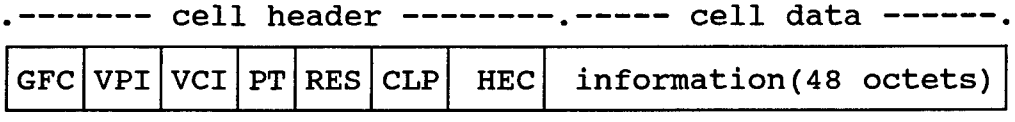
Interconnection between these two dissimilar groups requires careful consideration so that performance does not revert to the

least common denominator (ie low capacity and low adaptability). ATM represents the highest in link capacity, while STIP represents the ultimate in adaptability to a stressed environment.

An underlying concept of the global network is that the National Command Authority (NCA) will be able to communicate directly with personnel in the tactical environment. In order that graphics and video information be delivered in a timely manner, high capacity links are essential. ATM is a candidate for the wide-area part of this network since it is wideband and is being developed commercially. However, since ATM is designed for peacetime (well-behaved) use, STIP-like protocols must be incorporated to provide for survivability in the tactical area. This report documents research by the author that formulates methods which combine the advantages of both the ATM and STIP protocols into a network suitable for the military environment.

ATM NODE ARCHITECTURE

The ATM switch has N input FIFO buffers, each associated with an incoming link, and L output FIFO buffers for the outgoing links as shown in Figure 1. All input and output data is in the form of 53 octet ATM cells [4] with a 5 octet header as shown below.



GFC	Generic Flow Control(UNI only)	4 bits
VPI	Virtual Path Indicator	12bits
VCI	Virtual Circuit Indicator	16bits
PT	Payload Type	2 bits
RES	Reserved	1 bit
CLP	Cell Loss Priority	1 bit
HEC	Header Error Control	8 bits

The ATM switch routes the cell based on the Virtual Path Identifier (VPI) field in the header. An Internal Routing Lookup Table maps each active VPI number to one of the output buffers

(links). Figure 1 illustrates how the switch appends an internal node tag to each cell. The node tag causes the cell to take an appropriate path through the switching fabric to the desired output link. Thus the lookup table contents and the VPI field define a virtual circuit route through the switch for all incoming cells.

The control processor has the important function of updating the routing table. It does so by accepting requests to set up a Virtual Path Connection (VPC) from another switch via the control input buffer. In turn the controller can send requests to other switches via the controller output buffer. The physical path through which cells belonging to a session flow is determined and fixed for the duration of the session through a call set-up procedure. The control processor generates updates at a relatively low rate compared to the 150 Mbps rate at which individual cells are routed through the switch.

Figure 2 shows how a VPC is set up between source T and destination H. The routing tables at each intermediate ATM nodes define the VPC. The sequence of cells generated by each node during one transit of the VPC is shown. For clairty only the VPI is shown in the header along with the data D.

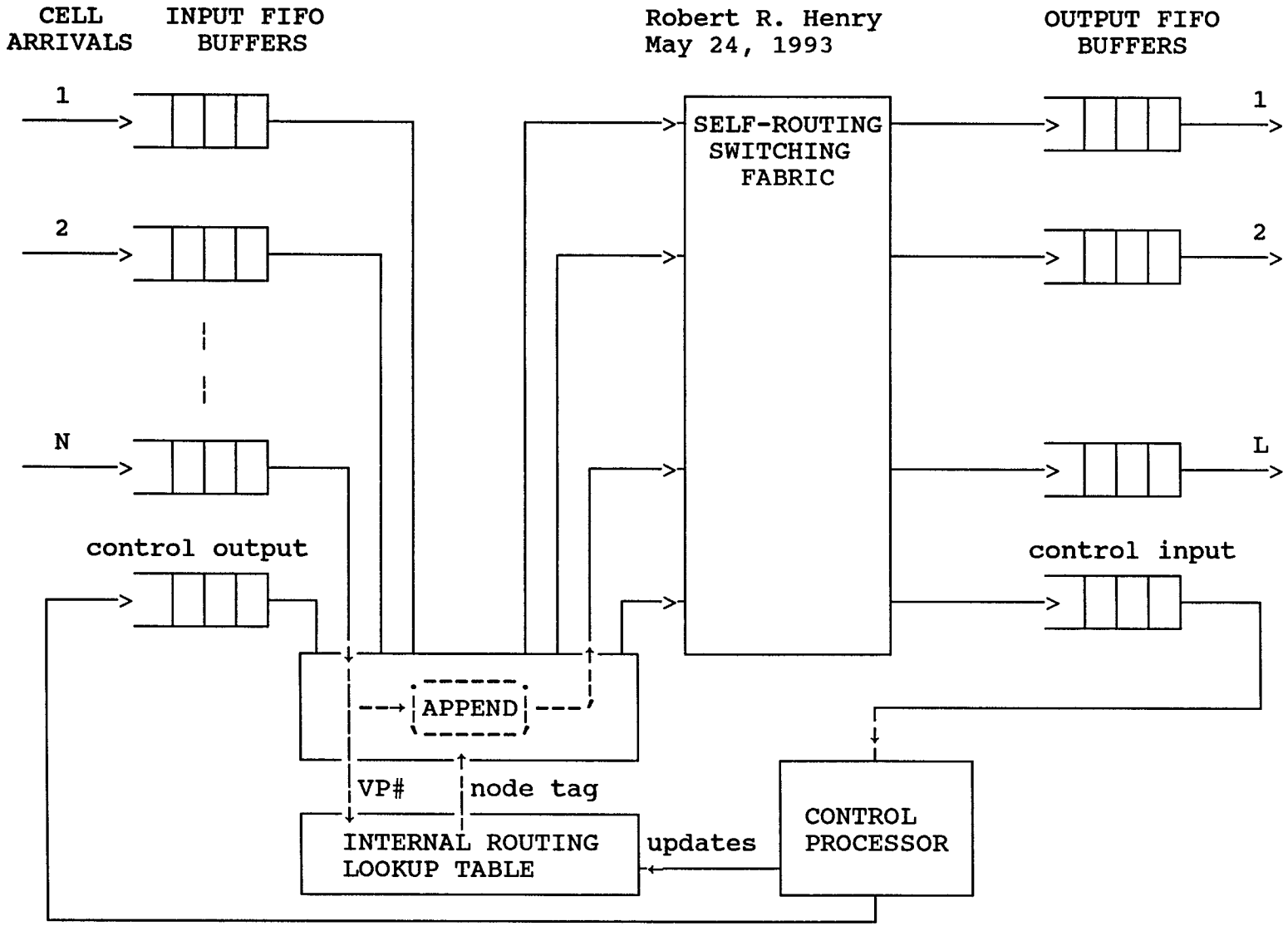


Figure 1. ATM Node i Switch Architecture.

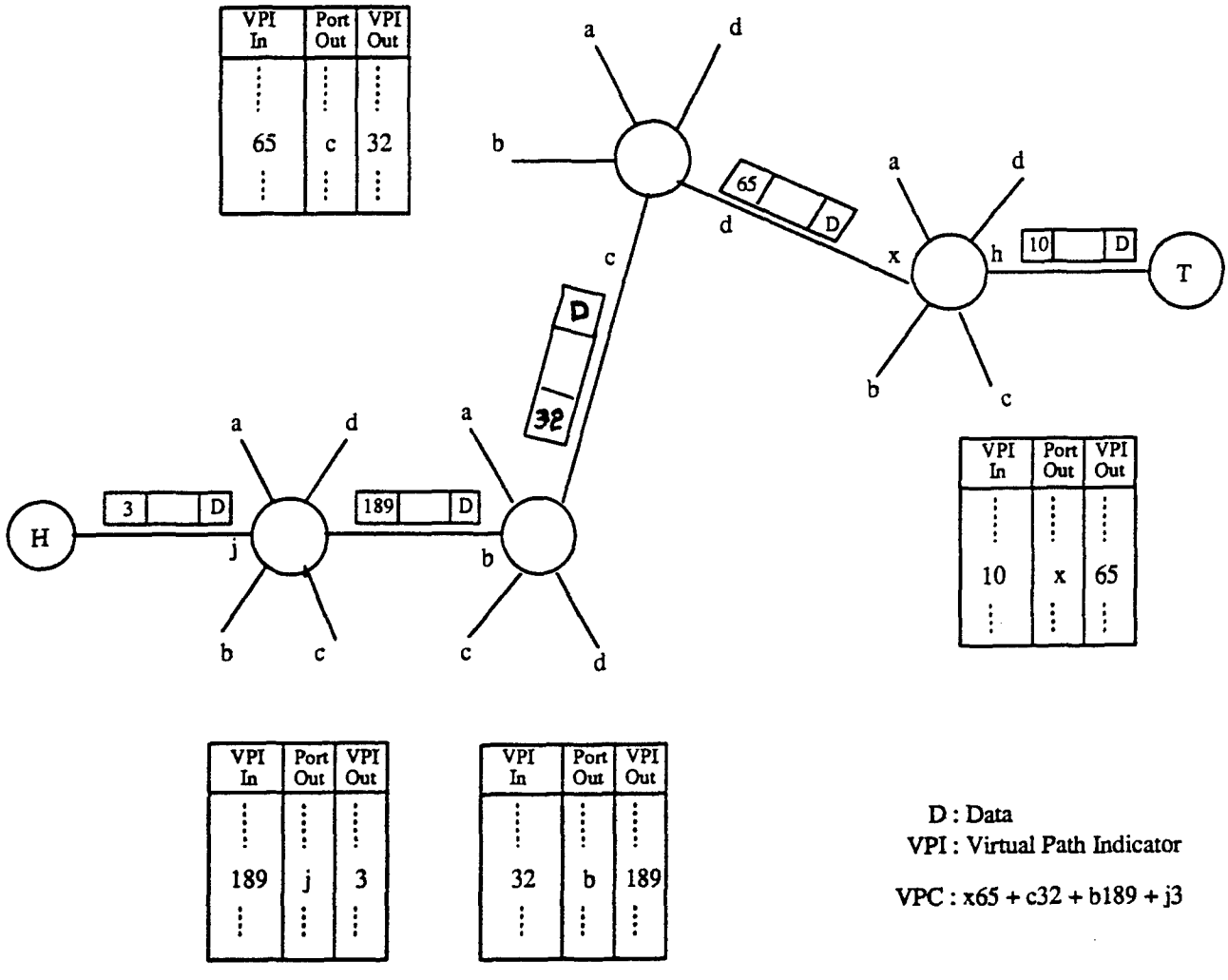
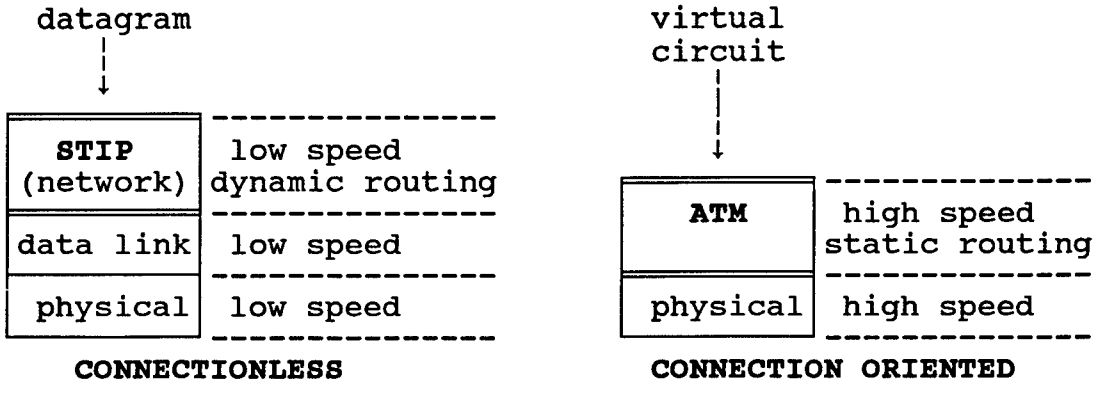


Figure 2. ATM Virtual Path Connection (VPC)

PROTOCOL COMPARISON

A comparison of the protocol stacks for both the ATM and STIP protocols is shown below, along with the layer performance characteristics, and the service provided to the layer above. The basic dichotomy is apparent - STIP provides datagram service which is low speed with dynamic routing; while ATM provides virtual circuit service at high speed with static routing. In addition, ATM routing is provided at a lower layer than is the more traditional STIP network layer routing.



The STIP protocol is a Connectionless (CL) procedure, while the ATM network is Connection Oriented (CO). There are significant differences in how each of these two methods accomplish data transfer, and this is summarized below.

ATTRIBUTE	CONNECTIONLESS	CONNECTION ORIENTED
Network	STIP	ATM
Connection	None	VPC
Data Unit	Datagram	Cell
Addressing	DA in header	VPI/VCI in header
Data Unit Route	Varies	fixed at setup
Node Storage	store DG	buffer a few cells

Table 1. Attributes of Connectionless and Connection Oriented Procedures.

This research formulates a way to incorporate the dynamic datagram routing paradigm into the high speed virtual circuit oriented ATM network. Both the Connection Oriented and Connectionless cases are considered.

CONNECTION ORIENTED DYNAMIC ROUTING

Circuit-Switched Routing

Interexchange Carrier (IXC) Circuit-switched networks have historically used static routing strategies, but in the last decade dynamic strategies have appeared. These new strategies fall into two categories - time dependent and state dependent. Each strategy attempts to find an alternate route for a call if no direct route is available. The alternate routes are limited to two-hops maximum. Some strategies employ a trunk reservation scheme to prevent the network from entering a state where a large portion of the network is allocated to two-hop, alternate paths and thereby blocking the more efficient direct paths.

AT&T's Dynamic Non-hierarchial Routing (DNHR) [4] is an example of time dependent routing. Data on the state of the network is collected over a period of time and used to forecast future expected traffic demands. A global optimization is performed on the network for certain time periods during the day based on historical data. Routing tables with up to 14 alternate, two-hop paths are downloaded to each switch. If a call can not be set-up on the first alternate path then the second is tried and so on. Trunk reservation is employed in DNHR.

Northern Telecom's Dynamic Traffic Management (DTM) [5] is an example of a centralized, state dependent strategy. The number of idle trunks, traffic that overflowed the direct route, and the CPU occupancy of each switch is reported to a central network manager every 10 seconds. The network manager calculates trunk reservation and call admission pacing parameters and downloads the data to the switch. Only two-hop alternates are allowed. AT&T's Trunk Status Map Routing (TSRM), France's System

to Test Adaptive Routing (STAR), the Bell operating company's Dynamic Routing-5 (DR-5) [6] also use centralized, state dependent routing.

British Telecom's Dynamic Adaptive Routing (DAR) [7] is an example of an distributed, state dependent strategy. Switches using DAR have one alternate route that all calls overflowing the direct route must take. If the alternate path is also blocked the call is lost, but a new alternate path is chosen at random from all possible two-hop routes. Trunk reservation is applied to the alternate routes. Japan's State and Time Dependent Routing (STR) is similar but restricts alternates to a set of optimum paths.

ATM Routing

The Virtual Path Connection (VPC) and Virtual Circuit Connection (VCC) in the ATM network can be viewed as being analogous to the switched circuit in the IXC network. Then a potential ATM dynamic routing paradigm is obtained by replacing the IXC network switched circuit by an ATM network VPC/VCC. Thus the set of two-link alternate routes may now be composed of elements with a variable number of concatenated links. This provides additional flexibility since the VPC/VCC need not be restricted to two hops.

The set of alternative routes are ranked according to a metric such as available bandwidth, congestion, delay, etc. A proposed metric based on survivability would yield alternate paths which are physically disjoint.

Dynamic routing may be implemented at the network edges (end-to-end) or at network intermediate nodes (internal). Internal routing responds faster to node/link failures than does end-to-end routing, but requires additional node intelligence. In addition the alternate routes may be predetermined (reserved) or determined as needed (on demand). The reserved method provides rapid response but ties up network resources, while the slower on demand technique efficiently utilizes network

resources. Table 2 summarizes four dynamic routing methods identified by the investigator based on the above discussion.

	RESERVED	**ON DEMAND**
END-TO-END	IXC	
INTERNAL	research	research

Table 2. ATM Dynamic Routing Mechanisms.

The research of this project is centered on both reserved and on demand internal dynamic routing. Figure 3 shows the sequence of events that occur when a link, marked with an "X", fails. Upon detection of the port b link failure, the upstream node sends a "reroute" cell out of port d to find a new downstream node. The VPI = 0 field routes this cell to the node control processor of the downstream node. Included in the cell is the destination address H, the required bandwidth B, and the VPI number to be used for the new route. This information is obtained from an augmented routing table as shown.

After acknowledgement, the upstream node updates its routing table by replacing the old entry with a new one which reflects the changes. In response the new downstream node updates its routing table and in turn sends out a "reroute" cell if it is not the destination node. Subsequent cells follow the new VPC.

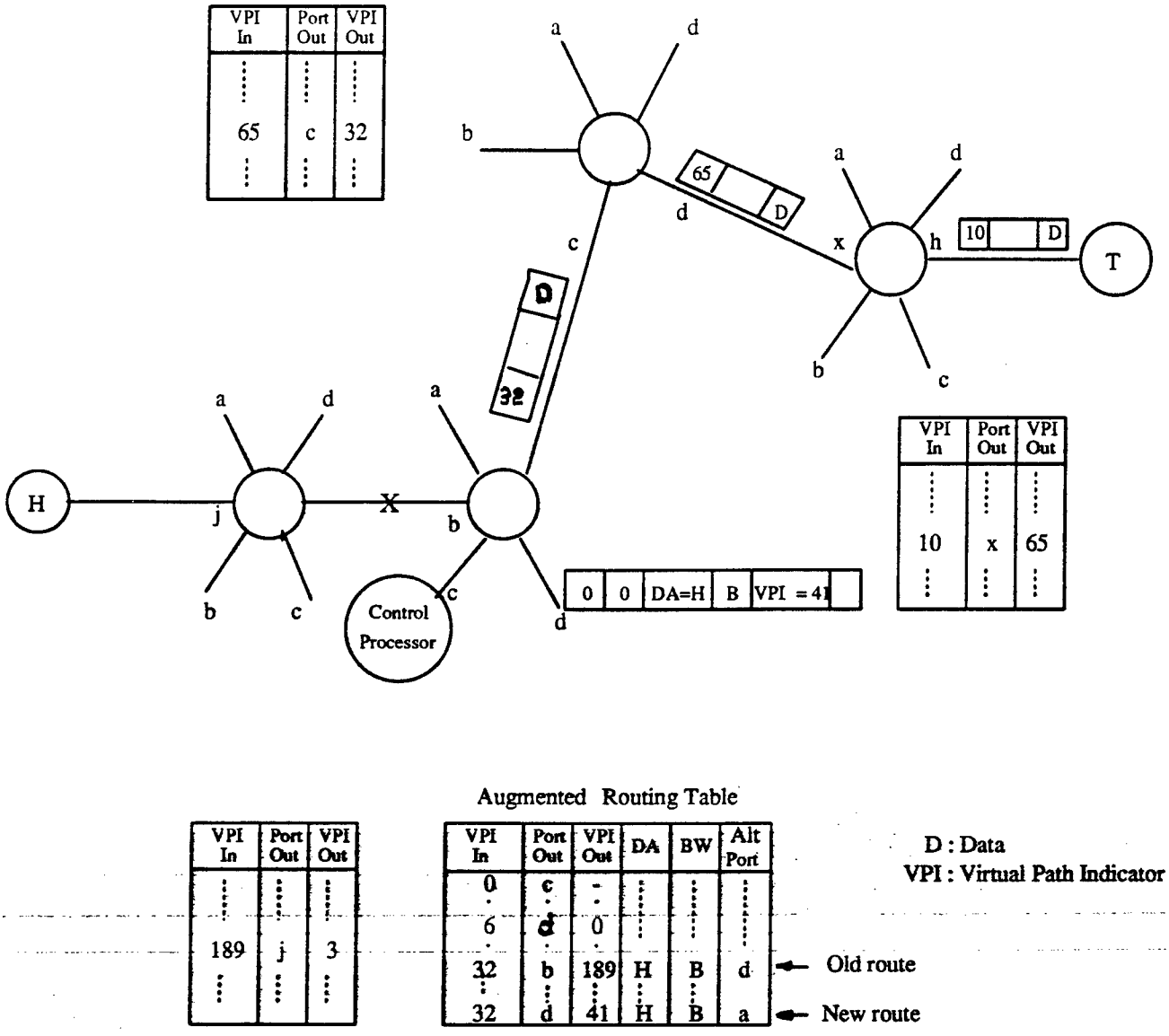


Figure 3. Proposed Internal Dynamic Re-routing Mechanism.

CONNECTIONLESS DYNAMIC ROUTING

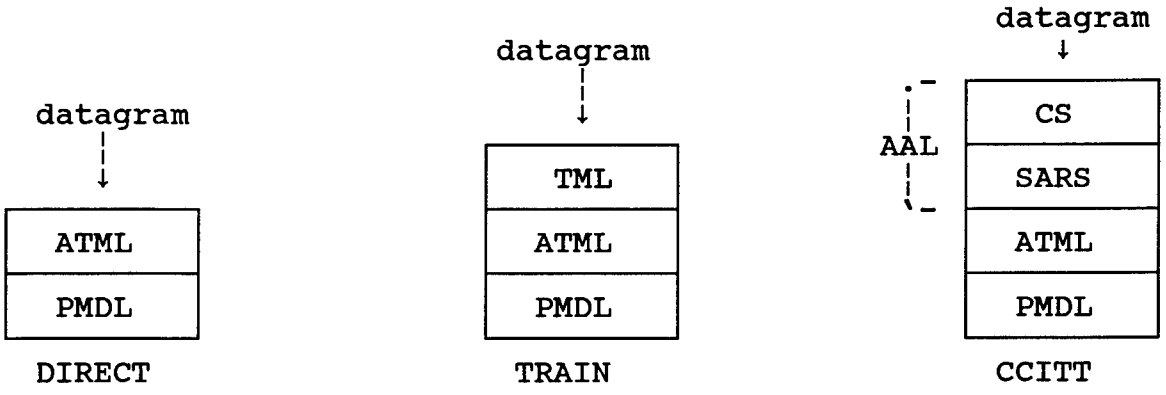
Datagram Dynamic Routing

Routing in datagram (connectionless) networks fall into two broad categories, shortest path/least cost and bifurcated [8]. Shortest path algorithms, such as the classic minimal spanning tree and Ford and Fulkerson's algorithm, find the optimum route for a single packet. Modified, distributed versions of Ford and Fulkerson's algorithm have been used in several networks. The unmodified distributed version is prone to looping problems. Many "cost" indicators have been used in implementations of these algorithms. Bifurcated routing minimizes the delay over the entire network. No commercial network has yet implemented this method; however, STIP is a bifurcated routing algorithm .

ATM Adaptation Layer Access

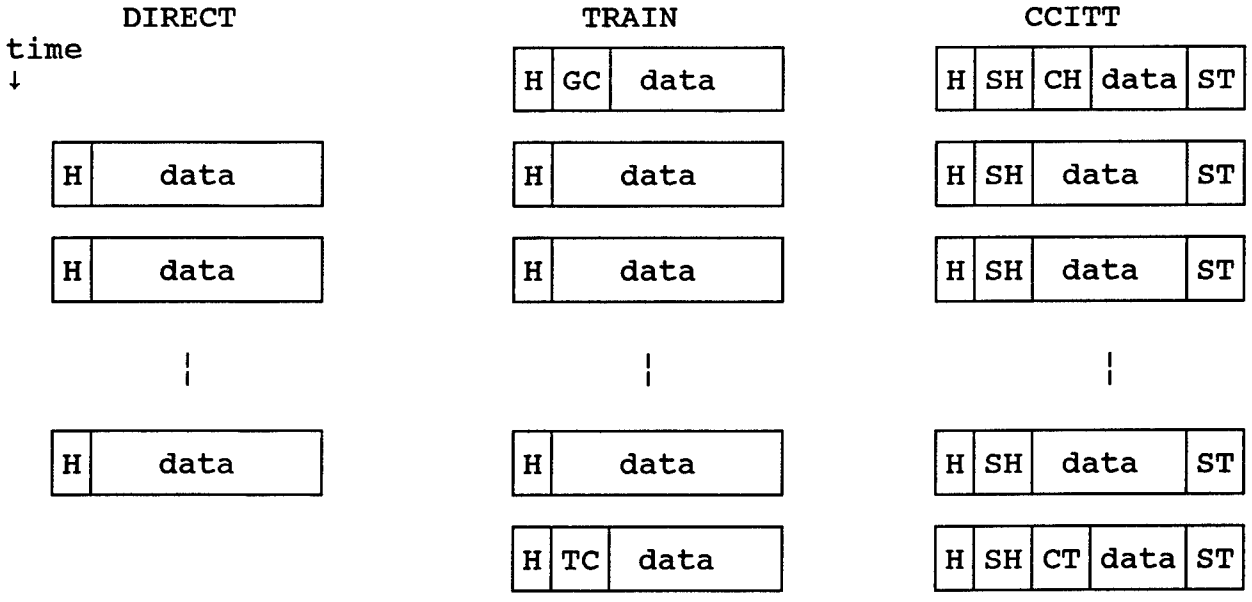
Efforts are currently underway by the CCITT [9] to provide datagram service via an ATM Adaptation Layer (AAL) that resides on top of the ATM Layer (ATML). This method embeds a 2-octet header and a 2-octet trailer in each cell data field to provide destination addressing, segmentation and reassembly, and error checking. The exact procedures are yet to be determined, but there is currently no effort to incorporate dynamic routing on a datagram basis. The protocol stack for this network is shown in Figure 4a. The corresponding sequence of cells generated by each datagram is given in Figure 4b.

Current research focuses on building a connectionless network "on top of" the ATM network using the services defined by the Adaptation Layer. Rahnema [10] proposes the use of this layer to establish permanent VC connections between pairs of connectionless networks. This method does not provide datagram service within the ATM network, but merely point-to-point connections between existing datagram networks.



ATML:ATM Layer TML:Train Mode Layer AAL:ATM Adaptation Layer
PMDL:Physical Media Dependent Layer CS:Convergence Sublayer
SARS:Segmentation & Reassembly Sublayer

Figure 4a. Protocol Stack for the Proposed ATM Datagram Access Networks.



H : ATM cell Header, 5 octets
SH : Segmentation & Reassembly Sublayer Header, 2 octets
ST : Segmentation & Reassembly Sublayer Trailer, 2 octets
CH : Convergence Sublayer Header, 4 octets
CT : Convergence Sublayer Trailer, 4 octets
GC : Guide Cell Header, 4 octets
TC : Trail Cell Header, 4 octets

Figure 4b. Datagram Cell Transmission Sequence for Proposed Networks.

A centralized connectionless server which receives datagrams from remote nodes is proposed by DePrycker [11]. Such a network has a "star" configuration with the disadvantage that the hub is a single-point failure and a prone to congestion. A distributed version in which connectionless servers are placed at each ATM node is suggested by Landegem and Peschi [12]. Each server uses the adaptation layer to exchange datagrams with other nodes. However the authors do not give details of how routing may be accomplished. We propose to establish a "Packet Switched Subnetwork" (PSS) with "Packet Handlers" located at strategically selected ATM nodes as shown in Figure 5. They would be selected based on criteria such as the number of hops, physical diversity, etc. The PSS will have the ability to reconfigure itself in response to failures. Thus the subset of ATM nodes which are Packet Handlers will change with time. They will be interconnected via VPC/VCCs on a reserved and/or demand basis. Datagrams are exchanged between Packet Handlers using the ATM Adaptation Layer (AAL) Type D service.

ATM Layer Access

In addition to the CCITT method, two other proposed methods for datagram access are illustrated in Figure 4. The "Direct" method inputs datagram octets directly to the ATM layer. There is a one-to-one correspondence between a datagram and ATM cell; thus the maximum datagram size then is 48 octets. A subset of VPI/VCI numbers are reserved for use as a datagram address field. The ATM switch would route these "datagram" cells separately from the standard VC cells.

The "Train" method uses a Train Mode Layer (TML) to interface between the datagram service and the ATM layer. For each datagram the TML generates a "train" of cells beginning with a Guide Cell, followed by Data Cells, and ending with a Trail Cell. The Guide Cell contains the routing information and leads the following "train" of cells through the network. The Trail Cell then deactivates the track established by the Guide Cell, thereby releasing switch resources for other datagram transmission. If

congestion is encountered at a node the Guide Cell and following Data Cells are stored in a datagram buffer until an alternate route becomes available.

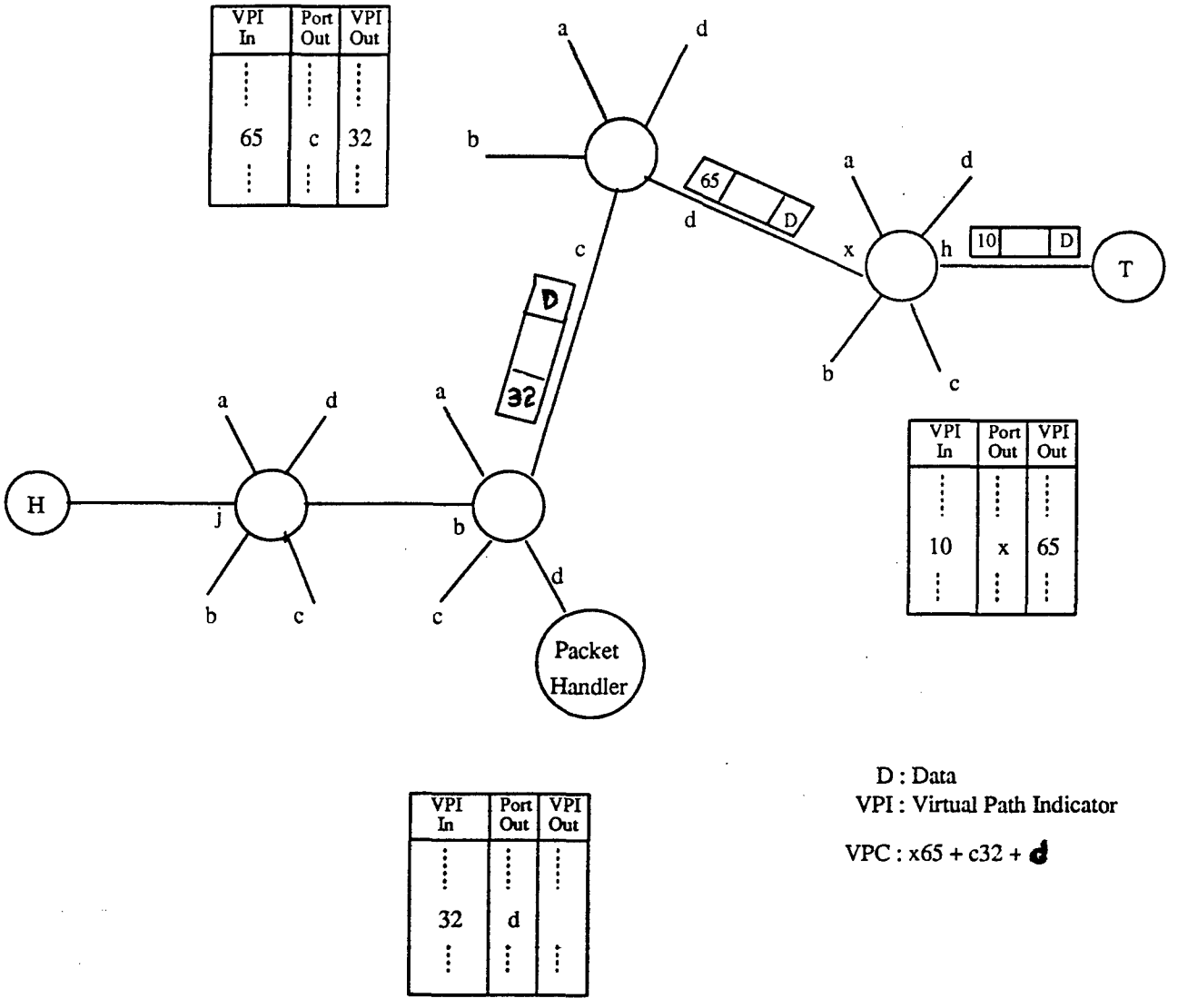


Figure 5. ATM Packet Switched Subnetwork (PSS).

DETECTION OF LINK FAILURES

Inherent in the operation of dynamic routing methods is the ability to detect link failures. It is important that detection occur rapidly so rerouting can be effected in a timely manner. The ATM Physical Layer SONET frame contains information that can be used to determine the status of a link [13]. The STS Path Trace octet is used to transmit a 64-byte, fixed length string repetitively so that a path-receiving terminal can verify its continued connection to the intended transmitter. The Path Status octet is allocated to convey back to the originating path terminating equipment the path terminating status and performance.

The importance of detecting errors as quickly as possible stems from the potentially high cell loss that may be incurred if rerouting is delayed. The minimum time to react to a condition within the network causing cell loss is the time for the condition to propagate back to the neighboring nodes plus the SONET frame length. This is because failure will occur at some point in the frame between the point in the current frame where failures can be detected and the same point in the next frame. Cell loss will be greater if error detection and recovery is placed at the end nodes rather than at each intermediate node. If an intermediate node fails between two end nodes 5000 km distant from each other and transmitting at the OC-3 rate of 155.52 Mbits/s, 830 Kbytes of data would be lost before the failure could even be detected. If rerouting could be performed at the intermediate nodes, with nodes being 500 km distant for example, the cell loss would be reduced to 170 Kbytes of data. Higher link speeds and delay within the nodes to react to network problems will increase the potential loss of data.

RESULTS

Very little has been done by other researchers to understand how ATM networks may be made to operate in a dynamic environment. The wisdom is that ATM networks will be well-behaved and operate over reliable physical systems. Much of the current effort is in the areas of ATM switch architecture development and call admission for VCs. Some attention has been given to offering connectionless (packet switched) services over ATM networks but this work lags well behind the work on developing connection oriented (VC) services. This perception was validated when Dr. Henry attended the 1993 IEEE INFOCOM in San Francisco.

ATM can be adapted to offer datagram (connectionless) services even though ATM is inherently a connection oriented service. The CCITT and other researchers have addressed this issue. This research proposes a packet switched subnetwork to provide connectionless service that will utilize and coexist with the commercial ATM network. This proposal utilizes VCs allocated on a semipermanent basis as the links between packet switching nodes. Additional intelligence at select ATM nodes performs the routing function. Routing will be done on a decentralized basis to avoid the single point-of-failure potential of centralized routing and network management. A variety of decentralized routing algorithms such as STIP, Internet, MILNET, etc. may be adapted to the requirements of a dynamic ATM network.

Propagation delay becomes a significant factor in wideband, high capacity networks. End-to-end routing is simpler to implement but is susceptible to much greater cell loss due to changes in the state of the network than if internal routing is used. Even with internal routing, cell losses may be non-trivial and presents an important area for additional research for those designing protocols for dynamic, ATM networks.

Five tasks were defined in the Statement of Work in the

Research Initiation Proposal and are listed below. The results and accomplishments for each task are summarized.

Task 1: Continue literature search for relevant developments.

The research has identified numerous references relevant to this research in the technical literature. Selected references may be found at the end of this report.

Task 2: Monitor industry progress in the ATM implementations.

Dr. Henry attended the 1993 IEEE INFOCOM held in San Francisco on March 30 through April 1. He discussed the latest developments in the field of ATM-based communication networks with the experts in the field.

Task 3: Continue research on how Datagram (dynamic) service can coexist with VC (static) service.

The research has identified how datagram service may be offered over a connection oriented (VC), ATM-based network. A Packet Switched Subnetwork is proposed that uses virtual paths and virtual circuits as the "links" that will carry dynamically routed packets. This method requires no modification to the ATM cell format as set by the CCITT and will coexist with commercial networks as currently envisioned.

Task 4: Research ATM cell transmission format for datagram service.

Datagram service can be implemented over ATM-based networks and three possible formats are suggested: the DIRECT, TRAIN, AND CCITT techniques proposed in Figure 3b. These methods are compatible with the envisioned ATM switch architectures and function by making appropriate updates to the ATM switch's routing table as demonstrated in Figure 4.

Task 5: Evaluate end-to-end and node routing of Datagrams.

This research suggests the use of a distributed routing as a

means to enhance ATM networks for the dynamic theater environment. The trade-off between end-to-end routing of cells as currently envisioned and routing at the nodes is that the network will be able to react much more quickly to changes in the state of the network at the cost of greater network complexity.

CONCLUSIONS

Most of the research surveyed in the literature assumes that ATM networks will have very low bit error and burst error rates and that the networks will be well-behaved. These assumptions are appropriate for static, commercial networks based on optical fiber links but are not valid for the dynamic theater environment. This research has identified several methods that can adapt ATM to the dynamic theater environment and concludes that a rapid response to exceptions is essential. Optimization of the methods identified will depend on a more specific understanding of the dynamic environment itself. The research has identified areas in which additional research is likely to be productive - specification of dynamic protocols and computer modelling of ATM networks.

The research has provided insight into the granularity of routing decisions noting that routing may be performed once for each call (session) or on a cell-by-cell basis or anywhere in-between. One of the areas identified for future research is that of finding the optimal point at which to make routing decisions in dynamic ATM networks. For instance, this research has shown how a VC call may be rerouted in progress to minimize the impact on the end users. The traditional approach for VCs is to break the connection and set up a new call when network problems are encountered.

The research has considered three methods for implementing

datagram service over ATM networks: the DIRECT, TRAIN, and CCITT methods. Each of these methods are compatible with the currently envisioned ATM architecture and function by updating the routing tables within the switches to alter the physical paths that the cells take. The most appropriate method for dynamic ATM networks is open to further study and is subject to hardware capabilities and conditions within the dynamic theater environment.

REFERENCES

1. "Evaluation and Development of Multimedia Networks in Dynamic Stress," Semiannual Project Report, SRI contract with Rome Laboratory, April 5-9, 1992.

2. "Secure Survivable Communications Network," User Meeting Report, GTE Government Systems contract with Rome Laboratory, July 15-16,1992.

3. Draft CCITT Recommendation I.150, "B_ISDN ATM Functional Characteristics," 1990.

4. Gerald R. Ash, "Design and Control of Networks with Dynamic Nonhierarchial Routing," IEEE Communications, Vol. 29, No. 10, October 1990, 34-40.

5. Jean Regnier and W. Hugh Cameron, "State Dependent Dynamic Traffic Management for Telephone Networks," IEEE Communications, Vol. 29, No. 10, October 1990, 42-53.

6. Prosper Chemouil, Janusz Filipiak, Paul Gautier, "Performance Issues in the Design of Dynamically Controlled Circuit-Switched Networks," IEEE Communications, Vol. 29, No. 10, October 1990, 90-95.

7. Peter B. Key and Graham A. Cope, "Distributed Dynamic Routing Schemes," IEEE Comuunicaitons, Vol. 29, No. 10, October 1990, 54-64.

8. Mischa Schwartz, Telecommunication Networks: Protocols, Modeling and Analysis, Addison-Wesley Publishing Company, Reading MA, 1988, pp. 259-327.

9. Draft CCITT Recommendation I.362, "B_ISDN ATM Adaptation Layer Functional Description," 1990.

10. M. Rahnema, "Frame Relaying and the Fast Packet Switching Concepts and Issues," IEEE Network Magazine, July 1991, pp18-23.

11. M. De Prycker, "ATM Switching on Demand," IEEE Network Magazine, March 1992, pp.25-28

12. T.V. Landegem and R. Peschi, "Managing a Connectionless Virtual Overlay Network On Top of an ATM Network," ICC '91, Conference Record, June 1991, pp.31.5.1-31.5.5.

13. Daniel Minoli, Enterprise Networking: Fractional T1 to SONET, Frame Relay to BISDN, Artech House, Norwood, MA, 1993, p. 478-479.

APPENDIX A

CELL LOST DUE TO TRANSMISSION LINK
AND NODE FAILURES

A model of a bidirectional virtual path through three nodes labeled A, B and A′ is presented for the purpose of ascertaining cell loss before a failure can be detected. This model estimates the number of cells that are lost due to a network failure before detection of the failure can occur. The reaction time of the network to the fault is not considered and is here assumed to be zero. Only the physical links for the virtual path of interest are shown, which in this model are unidirectional, single optical fibers identified by lower case letters. Nodes are identified with upper case letters.

Three failure modes are presented in which fiber 'a' fails at the points labeled x and y of Figure A-1 and a transit node failure characterized by node 'B' failing. It is assumed that failures are detected using the SONET path overhead. The model assumes a worst case scenario where the failure occurs immediately after a point in the SONET frame (125 μsec duration) where failures can be detected. This assumption adds an additional 125 μsec delay to the detection of the fault.

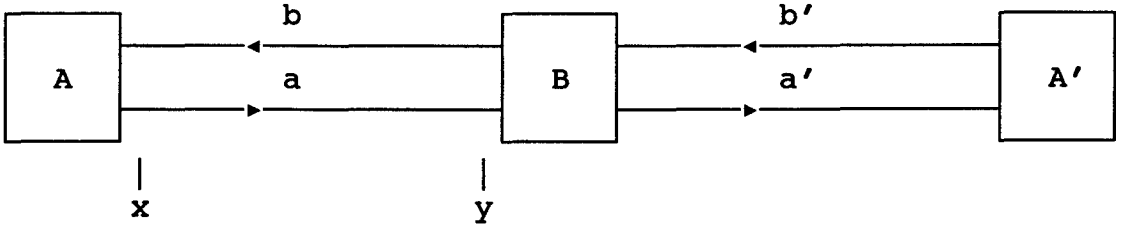


Figure A-1. Link and Node Failure Model

Transmission Rate:	D	bits/s
Propagation velocity:	V	m/s
Cell Loss:	CL	cells (53 octets)
transmission line distance:	d	m
SONET frame duration:	t_s	s
Time to detect fault:	T_1	s

Synchronous Payload Envelope of an OC-N signal.

$$D = 49.536 \times N \ Mbits,$$

Propagation Delay.

$$t_p = \frac{d}{V}$$

Time to detect fault at position x.

$$T_1 = 2t_p + t_s$$
$$= \frac{2d}{V} + 1.25 \times 10^{-4}$$

Time to detect fault at position y.

$$T_1 = t_p + t_s$$
$$= \frac{d}{V} + 1.25 \times 10^{-4}$$

The following three cases assume a propagation velocity of .75c (2.25×10^8 m/s and an OC-3 signal (148.608 Mbits/s data carrying capacity):

Mode 1: Node B fails

1) A and A′ continue to transmit unaware that B is not receiving
2) Cells on a and a′ are lost (same as if fiber had failed at point y)
3) Cells transmitted by A and A′ after the fault are lost

$$CL = 2t_p \frac{D}{424} + 2T_1 \frac{D}{424}$$
$$= \left(\frac{2d}{V} + 1.25 \times 10^{-4}\right) \frac{2D}{424}$$
$$= 6.231 \ cells/km + 87.6 \ cell$$

Mode 2: Fiber a fails at x

1) Cells already on a are not lost
2) Cells transmitted by A after fault are lost

$$CL=(2t_p + t_s)\frac{D}{424}$$

$$=(\frac{2d}{V} + 1.25\times10^{-4})\frac{D}{424}$$

$$=3.115\ cells/km + 43.8\ cell$$

Mode 3: Fiber a fails at y

1) Cells already on a are lost
2) Cells transmitted by A after fault are lost

$$CL = \frac{t_pD}{424} + \frac{T_1D}{V}$$

$$=(\frac{2d}{V} + 1.25\times10^{-4})$$

$$=3.115\ cells/km$$

A receiving node will in actuality run some fault detection algorithm that will take an as of yet undetermined length of time. Additionally, the algorithm may dispatch an alert to the affected entities to inform then of the condition. The alerts may take on different levels. A low level alert may indicate a high number of errors in the SONET stream, which might be a precursor to a total failure. A midlevel (yellow) alert might be issued at the first indication that a failure has occurred and a high level (red) alert might be issued after one or more entire SONET frames are missing or unusable. SONET frames are also scrambled so the loss of even part of the frame results in the loss of more than just the data in the affected portion of the synchronous payload envelope (SPE). Even so, the number of cells lost on a faulty SONET frame is small compared to the potentially very large number of cells that may be in transit on a link. For a moderately long link of 500 km, a node failure would result in the loss of 3202 cells due to propagation delay while a link of 5000 km would lose 31,243 cells with a node failure. The failure of a single fiber would result in about one-half of these losses.

CONGESTION CONTROL FOR ATM NETWORK IN A TACTICAL THEATER ENVIRONMENT

Benjamin W. Hoe
Department of Electrical Engineering
Polytechnic University

Final Report for:
AFOSR Summer Research Extension Program
Rome Laboratory
Griffiss Air Force Base

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, DC

Contact Number: 93-001
October 1993

ABSTRACT

The congestion control in the ATM network is a critical technical issue that requires extensive simulations under different scenarios in addition to the theoretical derivation of closed form formulas and equations. In a tactical theater environment, this task is complicated by the tactical communications requirements: multi-level security, service priority levels, integration of tactical traffics, and rapid and effective response to the dynamic changes. This paper presents the results and the recommendations of the research work conducted in order to support the Telecommunications Division (C3D) of the Rome Laboratory effort in execution of the DDR&E Science and Technology Thrusts - Global Surveillance and Communications and Precision Strike. The results of this effort also related directly to both the "Global Grid" and "Artemis" Advanced Technology Demonstrations where the ATM is an important component. The study reveals that the closed form solution is possible for ATM switch with both the input and output queuing which is the congestion control technique used in the Secure Survivable Communications Network (SSCN) project. However, closed form solution for end-to-end network performance depends on the dimension and the distribution of network layout, and require additional study. The simulation model for node level analysis is discussed, and the future research works are identified and discussed.

Acknowledgment

This research work is sponsored by the Air Force Office of Scientific Research (AFOSR) under the Summer Research Extension Program (SREP).

I would like to express my deep appreciation to the Telecommunications Division of the Rome Laboratory (RL/C3D) for the supports that I have received during my two research assignments at the Rome Laboratory, Griffiss Air Force Base. My two years of association with the RL/C3D has greatly enhanced my technical expertise, and transformed me from a graduate student to a working engineer. My special thanks go to Nick Kowalchuk and Joseph Zdanowicz for their advises, guidelines, and constant supports. Without Nick and Joe, I'd be still wondering about what is the ATM!

I also thank Dr. John Evanowsky, and C3D division chief Daniel McAuliffe for their supports and endorsements for my research work. I am very grateful to Prisicilla, Serey, and other staff members of the Rome Laboratory for their helps which make this work possible. I shall also mention my colleague Joe Schlareth from the MITRE corporation for reviewing this final report.

CONGESTION CONTROL FOR ATM NETWORKS IN A TACTICAL THEATER ENVIRONMENT

Benjamin W. Hoe

1) Introduction:

This paper presents the analysis of congestion control techniques in tactical Asynchronous Transfer Mode (ATM) networks in order to support the Tactical Communications Land Combat Zone Post-2000 needs for integrated information: voice, data, video, and messages. The transmission speed for future Broadband Integrated Services Digital Network (BISDN) will start from 155.56 Mega bit per second (OC-3), and can increase beyond 1Giga bit per second since the fiber optic medium can provide high bandwidth with very low bit error rate. Such high speed network will simultaneously support different types of traffic with different characteristics, and different service requirements. The integration of loss sensitive information such as data, and delay sensitive information such as interactive video requires new queue management techniques to optimize the delay-loss performance. The congestion control in tactical ATM networks is complicated by this two dimension performance requirement, in addition to the military communications requirements: multilevel security, service priority level, and integration of low speed high bit error rate tactical traffic. There are some congestion control techniques for commercial ATM network proposed by different organizations. These techniques may not be effective in tactical theater environment, where traffic patterns will be ranged from few hundreds bit per second to 155.56 Mbps (OC-3), the speed expected to be used in POST 2000 Wide Area Network (WAN) architecture. Especially, loss of

some communications nodes and links, and service degradation due to enemy jamming add different requirements for congestion control in military networks.

The figure 1 illustrates the ATM network in POST 2000 architecture. It shows that ATM network must deal with variety of communications systems: voice networks, tactical networks, local area networks, high performance workstations, and video image transmission.

Successful implementation of ATM technology for BISDN is based on: 1)Switching 2)Transmission, and 3)Effective Congestion Control. The primary component of ATM network is an ATM switch which routes incoming ATM cell (53 bytes long packet) to its destination output. Some basic ATM switch architecture will be discussed in section 3. The transmission medium in ATM network is mainly optical fiber, which supports high speed transmission with very low bit error rate. Currently, there are five Giga bit testbeds sponsored by the National Science Foundation (NSF), and Defense Advanced Research Projects Agency (DARPA), in addition to the Department of Defense (DOD) sponsors Theater Extension Network (TENET). Congestion control, the center core of this paper, is an important issue still have to be solved in order to achieve the successful implementation of ATM network. GTE corporation has proposed to use an ATM switch with both input and output queues for the Air Force ATM testbed, which is being developed under the Secure Survivable Communications Networks (SSCN) project. The objective of this paper is to present the simulation model developed to perform congestion control study on above proposed ATM switch, discuss the performance of the output queuing technique against the proposed input/output queuing technique, and develop potential enhancements on current technology.

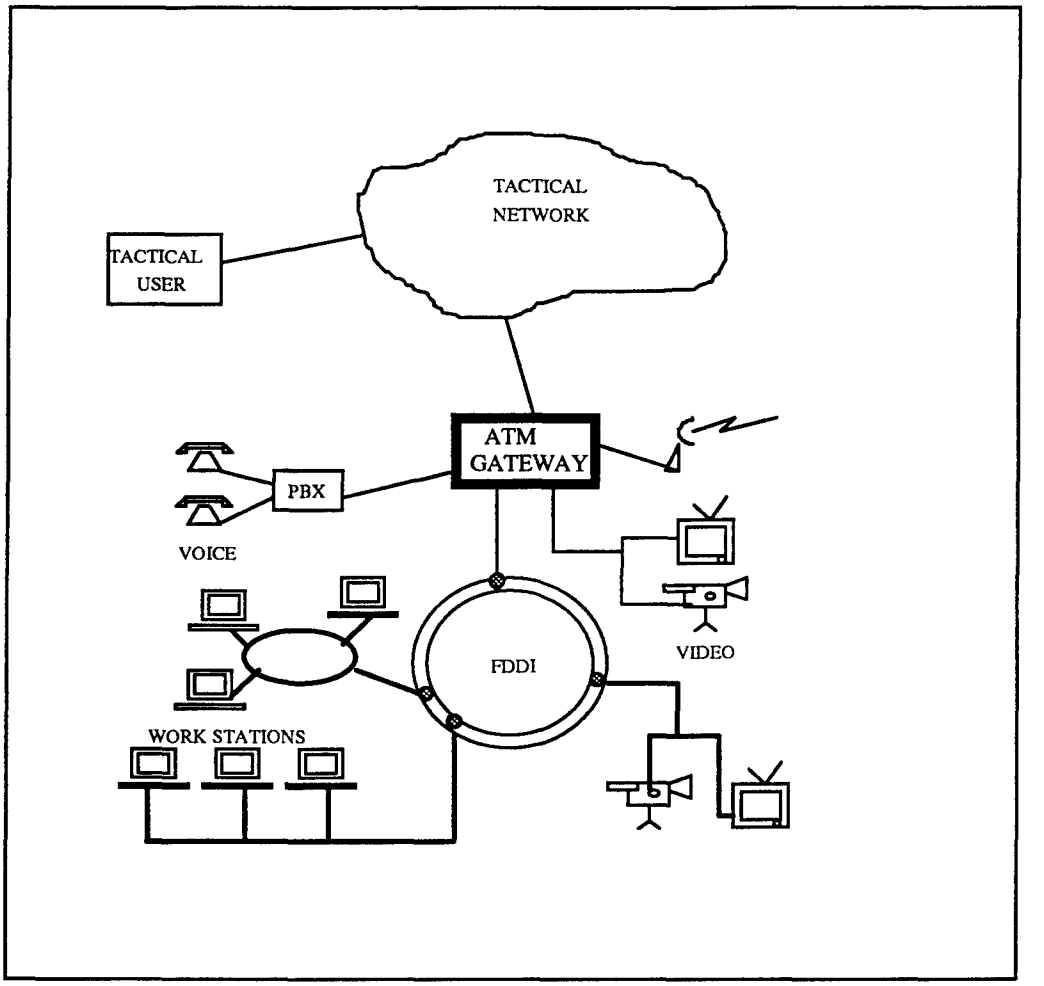


Figure 1. ATM Switch in NATO Post 2000 Tactical Communications Architecture

The research work discussed in this paper is originated from the work conducted at the Rome Laboratory in the summer of 1992. Brief background of the current work is discussed in the next section. The section 3 provides different switch architecture to establish basic ATM concepts before the discussion of queuing analyses in section 4. The section 5 describes the simulation models developed as the major effort of this research. All simulation models and source codes have been delivered to the Rome Laboratory. The section 6 presents two simulation outputs, and the section 7 discusses two recommendations derived from this research effort.

2) Background of Current Work:

The Rome Laboratory (RL) is constructing an ATM testbed to conduct the Advanced Technology Demonstrations (ATD) in order to support technologies critical to the future networking requirements of the Air Force. In this testbed, an ATM switch with both input and output queues are used as buffers for incoming traffic if a destination output port is busy or not available for service. Generally, the use of input queue reduces the throughput due to the head of the line (HOL) blocking. The use of output queue can achieve the optimal throughput-delay performance, however, cell loss cannot be avoided if congestion continues so the offered load (incoming traffic) exceeds the buffer capacity. In general, queue size is selected according to the steady state offered load such as $\lambda_{offer} < \lambda_{service\ rate}$ so congestion will be solved by itself in the steady state. Selection of queue size in a tactical network according to this rule can not guarantee no cell loss since the offered load λ_{offer} is difficult to determine due to the dynamic nature of traffic characteristics in military environment. The possible loss of some communications resources such as transmission links and switching nodes, and jamming contribute unusual heavier traffic loading on some links and nodes. Such dynamically changing situation in battlefield, and the Hot Spot effect (Appendix 1) which is highly possible in tactical environment make it very difficult to predict the steady state queue length and traffic loading. Hence, use of output queuing alone cannot guarantee the low cell loss probability (P_{loss}) in tactical ATM network.

The GTE proposed to use the input and output queuing technique, which was first presented by Iliadis and Denzel of IBM corporation [6]. It has been shown that appropriate use of input and output queues combination reduces the cell loss probability with high throughput capability. However, any use of input queues can cause HOL

blocking which can reduce the switch throughput.

The initial study on congestion control issues was conducted in RL and funded by the Air Force Office of Scientific Research (AFOSR) during the Summer of 1992 under SRP program. Then, RL has decided to build a simulation model for ATM switch with both input and output queues to study these traffic control issues. OPNET object oriented software was used as the design tool, and the result of this simulation model can be easily modified or incorporated with other design works performed by the RL and GTE. Three tasks identified under initial project are 1) Design and analysis of diverse ATM traffic in tactical environment 2) modeling and analysis of diverse transmission media in tactical environment and 3) defining threat scenarios. The task one is most difficult one since the integrated traffic patterns in BISDN, especially in military environment is still not well understood, while task two and task three are relatively well defined compare with task one. Some of the project related to these tasks are the Development of Multimedia Networks in Dynamic Stress (EDMUNDS) project sponsored by the DARPA, and the Communications Networks Operating System (CNOS II) project of the RL, Griffiss Air Force Base.

3) ATM Switch Architecture

The ATM network, BISDN with ATM protocol, transmits fixed size packet (53 Bytes) known as ATM cell over SONET medium with the speed of 155.56 Mbps (OC3 speed). The ATM switch which routes cells from input ports to output ports. During the routing of ATM cells from the input side to the output side of the ATM switch, there are two possible blockings-- internal blocking and output port blocking, as shown in figure 3. The internal blocking can be avoided by switch design and the output port blocking due

to output port contention is solved by contention resolution algorithms. Some of the well known internal non blocking ATM switch architecture are briefly discussed in section 3.1, 3.2, and 3.3 because analysis of congestion control and cell loss probability is related with the switch architecture. The contention resolution algorithms used to solve output port blocking are Recirculation Algorithm from AT&T, Three Phase Algorithm from Bellcore, and Ring Reservation Algorithm from Bellcore.

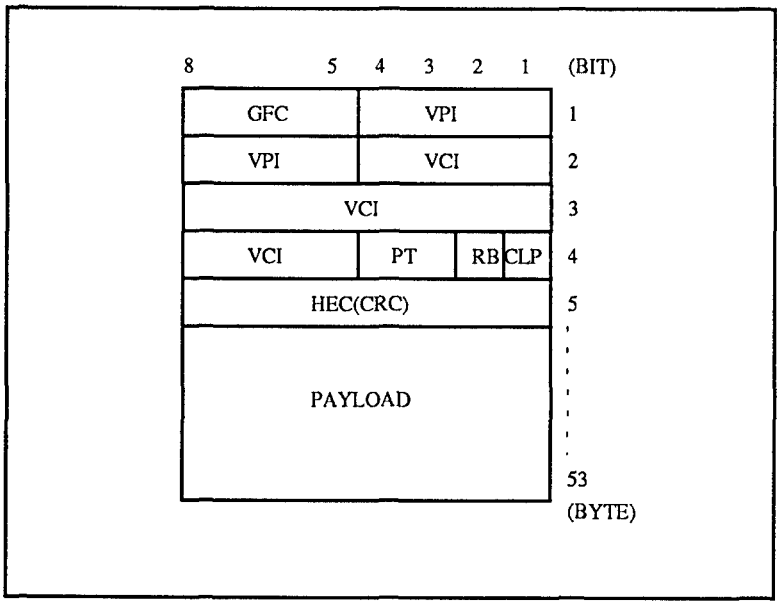


Figure 2. ATM Cell Format

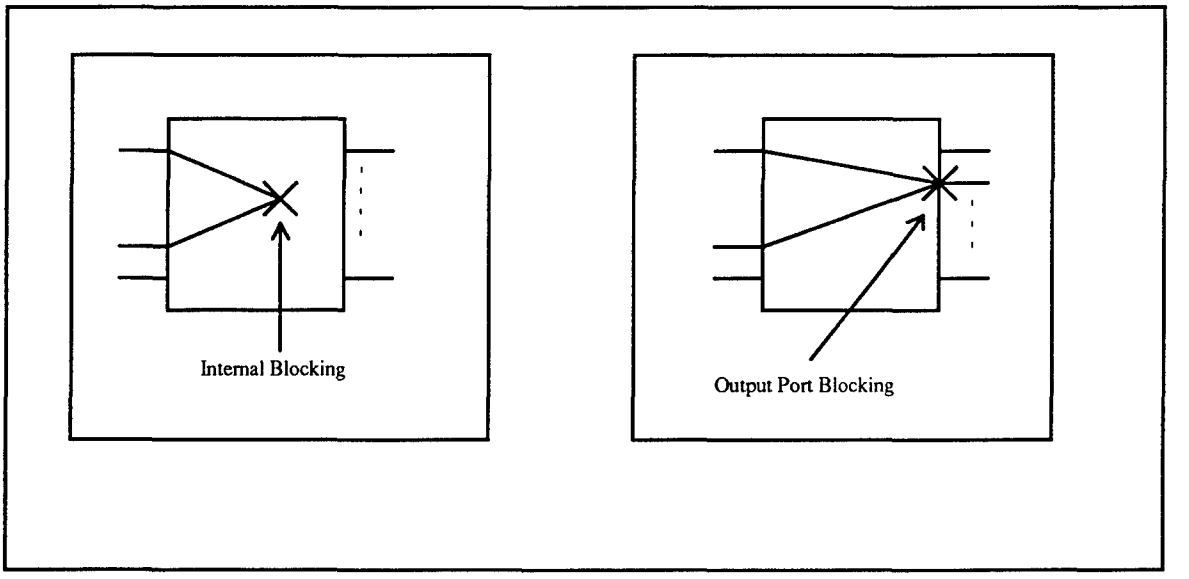


Figure 3. Internal Blocking in ATM switch caused by the same intermediate point taken by two inputs, and Output Port Blocking caused by two inputs with the same destination output.

3.1) Batcher-banyan Switch

The Batcher network is a sorting network, which sorts the inputs into monotonically increasing order. The Banyan network is a self routing network, which can route the cell to its destination using the address in the header field. The Banyan network is internally non-blocking if all inputs are compact and sorted in monotonically increasing order. Therefore, the combination of Batcher network and Banyan network, known as Batcher-banyan network is used as an non-blocking ATM switch. It has the complexity of $N(logN)^2$.

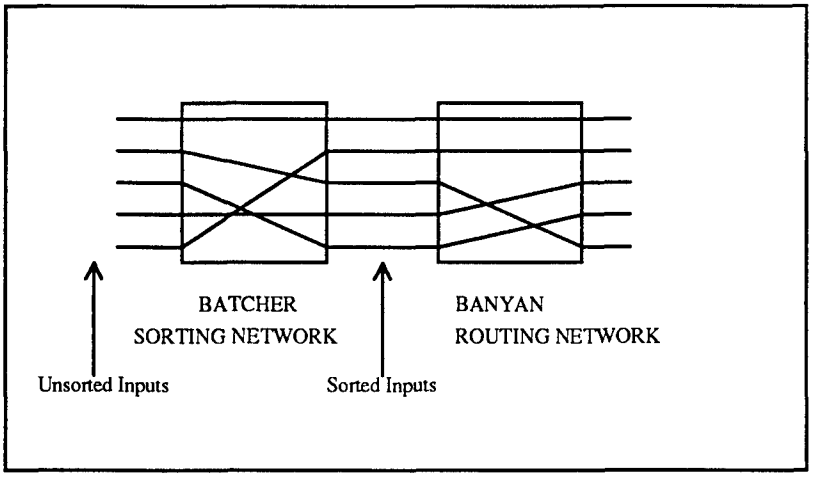


Figure 4. Batcher-banyan Internal Non-blocking Switch

3.2) Knockout Switch

The Knockout switch is another well known switch developed by the AT&T Bell Lab. In this switch architecture, all input lines are connected to all output ports through the bus interfaces which contain packet filters, *N x L* concentrator, and a shared buffer.

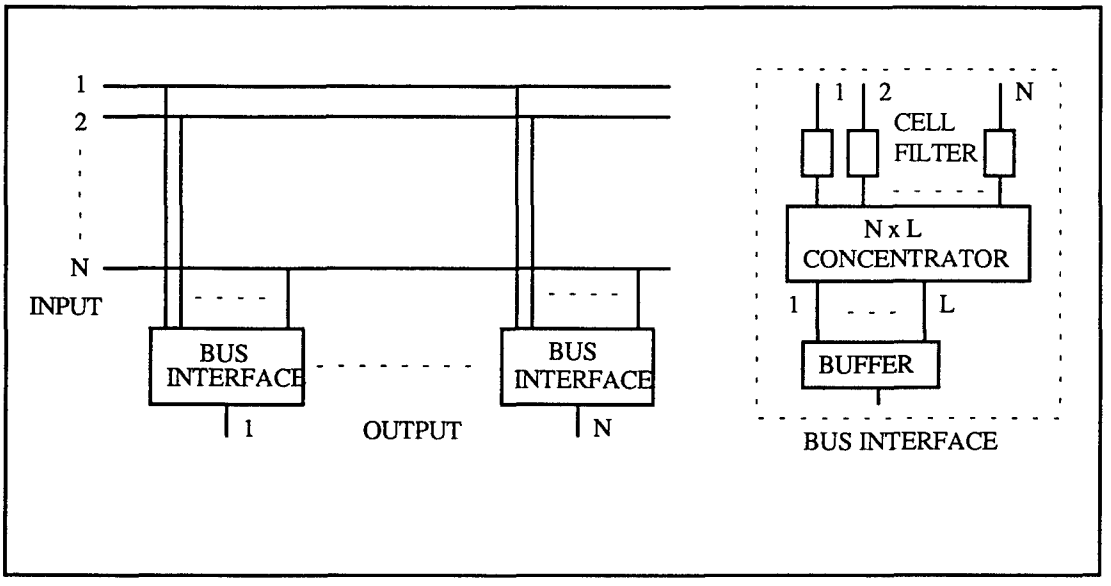


Figure 5. The Knockout Switch with *NxL* Concentrator

The original version of knockout switch uses output queuing. It has been shown that the cell loss probability P_{loss} of 10^{-6} can be achieved by selecting L=8 for large N. GTE is developing a tactical ATM switch based on this architecture with input and output queuing for the SSCN project. Use of knockout switch architecture in a large scale network may not be a good choice since its switch hardware complexity is in the order of N^2.

The Knockout switch is designed in the way that each incoming cell is broadcasted over N channels. Concentrators are connected to N channels through the cell filters which examine the output port address (destination address) of arriving cell, and only admit those cells that have the valid output port address into the NxL concentrator. For example, cell filters inside Bus Interface 1 admit cells with output port address 1 to the concentrator and reject all other cells with different output port address. Each concentrator inside each bus interface has N inputs and L outputs where L is less than N. Therefore, a concentrator can receive L packets simultaneously. If K cells destined to a same address for the case K is greater than L, L cells will go through the concentrator and the remaining K-L cells will be discarded. The selection of L packets can be prioritized inside the concentrator. The detail analysis of Knockout switch can be found in [9].

3.3) Dual Shuffle-Exchange Switch

This architecture is developed by S. Liew and T. Lee [8] from the Bellcore recently. Probably, it will be the best choice for a large scale ATM network since its complexity is *NlogN*, the Shannon's lower bound. This switch architecture is based on the concept that shuffle network and unshuffle network are mirror images of each other, and the deflection routing error caused by one network can be corrected by its mirror image

network.

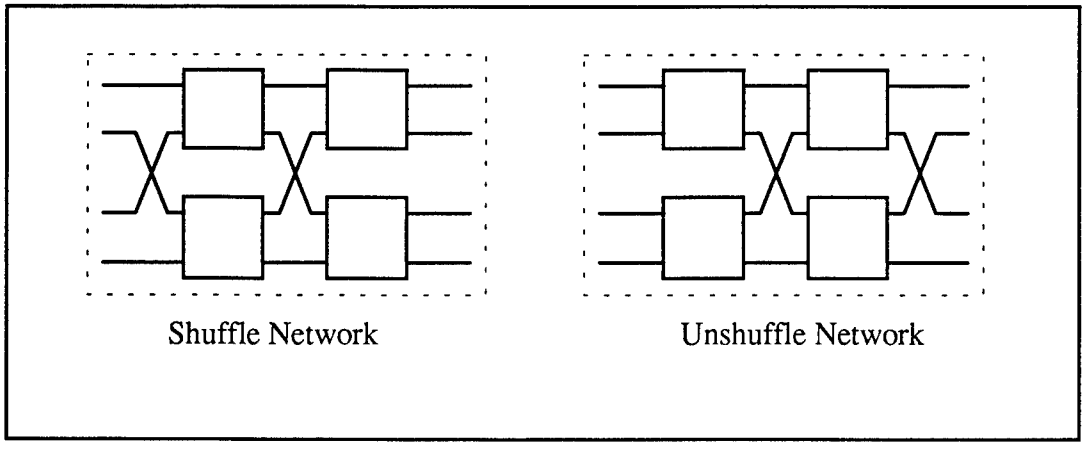


Figure 6. Dual Shuffle-Exchange Switch

3.4) Switch Architecture and Cells format used by GTE

The detail of ATM switch fabrication architecture and the cell format are proprietary information and will not be discussed in this paper. GTE switch is generally based on the Knockout switch [9] concept and used input and output queuing technique [6] for congestion control. The ATM cells arrived at input ports will be appended with node tag which contains priority information and security information. This 5 bytes node tag is stripped off at the output side of the switch to conform the standard ATM cell format.

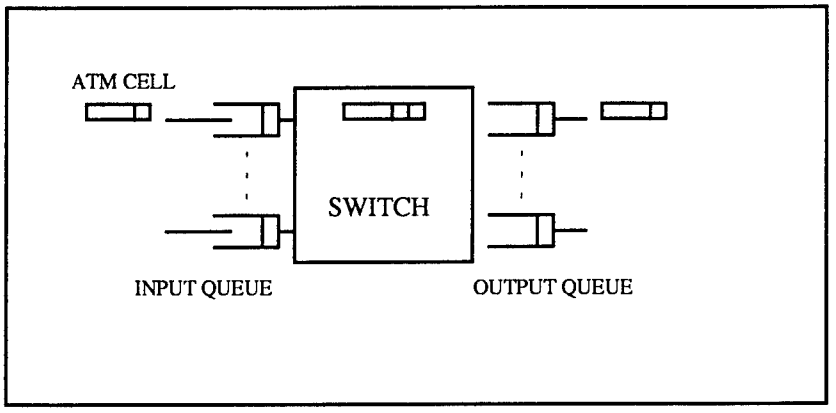


Figure 7. ATM Switch with Input and Output Queuing. Node tag is appended at the input side and stripped off at the output side

In addition to those basic architecture discussed in section 3.1 through 3.4, there are various versions of ATM switch architecture based on the banyan network such as Tandem-banyan architecture, Parallel-banyan architecture, and Dilated-banyan architecture. Section 3.0 through 3.4 are intended to provide insight into the ATM switch concepts and architecture as groundwork for discussion of queuing and congestion control analysis, the main objective of this paper. For detail on ATM switch, refer J. Hui [5], Yeh [9], Liew [8], and GTE proposal [2]. The following table summarizes the basis switch architecture, complexity, and associated queuing techniques.

Switch Architecture	Complexity	Queuing Technique
Batcher-banyan	$O(N\log^2 N)$	Input Queuing
Knockout	$O(N^2)$	Output Queuing
Dual Shuffle	O(NlogN)	Output Queuing
Tandem-banyan	$O(N\log^2 N)$	Output Queuing
Parallel-banyan	$O(N\log^2 N)$	Output Queuing
Dilated-banyan	O(NlogNlog(logN))	Output Queuing

Table 1. Switch architecture with respective complexity and queuing technique

4) Queuing Analysis

The function of an ATM switch is to route cells from input port to output port. If more than one input port have cells destined for the same output port, there will be only one contention winner and the rest must be stored in either input queue or output queue till the output port is available for next service. The offered load is selected to be smaller than the output port rate in order to preserve the stability in the queues, however, the offered load can exceed the service rate in worst case war time scenario. The cell loss will occur after the queues reach its capacity. Therefore, selection of optimal queue size is important; small queue will result in large cell loss and large queue size will result in higher hardware complexity.

4.1) ATM Switch with Input Queuing

The input queuing technique is illustrated in figure 8. If an arriving cell is destined to an output which is not serving any cell, then the arriving cell is switched to the destination output and no queuing is required. But, if multiple inputs have cells destined to the same output at the same time, then the output port can facilitate only one cell at a time, and contention losers must wait in the queue. Other situation is if an arriving cell is destined to an output which is serving another cell so the arriving cell must wait in queue for service. The major disadvantage with the input queuing technique is the concept known as the head of the line (HOL) blocking. As illustrated in figure 8, both cells at the head of the line of the Input 1 and Input N are destined to the same Output 3. The cell from Input 1 wins the contention so cell from Input N must wait at the HOL. Suppose a next arriving cell at the Input N is destined to Output 1, which is idle and available for service. But, this recently arrived cell can not be routed to its destination Output 1

because the HOL cell is blocking its way. This HOL blocking limits the switch throughput to 58% in switch with input queuing.

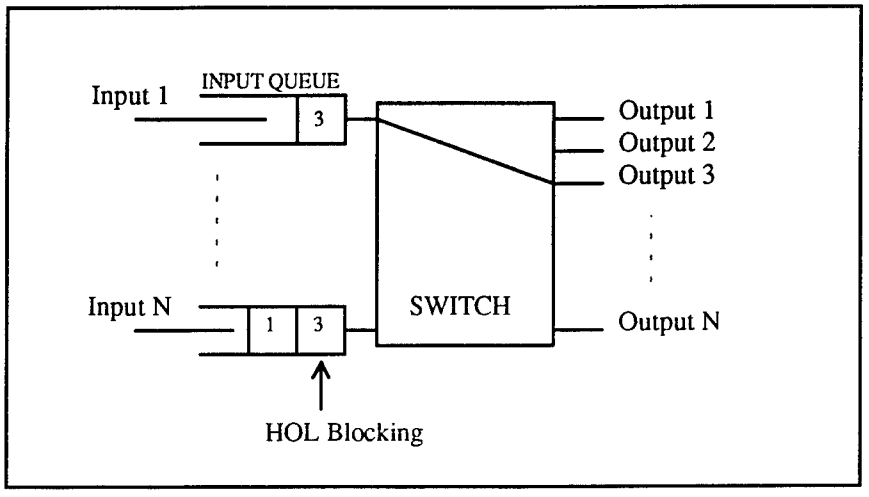


Figure 8. Switch with Input Queues

Consider a NxN switch with the following parameters for analysis of input queuing:

N_j : Total number of cells destined to a tagged *j* output in current time slot

N_u : Total number of unblocked inputs in next time slot

$\in (N_j)$: Number of cell successfully delivered to *j* output during current time slot

$\in (N_j) = \min (1, N_j)$

λ_i : Probability of a cell moves to HOL or Offered load at input port

λ_o : Throughput

The total number of unblocked inputs in the next time slot is just total number of input port minus total number of blocked input port in current time slot. Which is:

$$N_u = N - \sum_{j=1}^{N} [N_j - \in (N_j)] \qquad (1.1)$$

from equation (1.1), the expected value for N_j is obtained as:

$$E[N_j] = 1 + \lambda_o (1 - \frac{1}{\lambda_i}) \qquad (1.2)$$

Another equation for $E[N_j]$ can be derived from the following dynamic equation:

$$N'_j = N_j - \in (N_j) + A_j \qquad (1.3)$$

where

N'_j : Total number of cell destined to output j at next time slot

A_j : New arrival destined to output j

$$E[N_j] = \frac{\lambda_o (2 - \lambda_o)}{2(1 - \lambda_o)} \qquad (1.4)$$

From equation (1.2) and (1.4),

$$1 + \lambda_o (1 - \frac{1}{\lambda_i}) = \frac{\lambda_o (2 - \lambda_o)}{2(1 - \lambda_o)}$$

$$\lambda_i = \frac{2\lambda_o (1 - \lambda_o)}{2 - 2\lambda_o - \lambda_o^2} \qquad (1.5)$$

In order to analyze the maximum throughput, we assume that there is always a cell to be routed to output port at anytime, that is, λ_i is 1. Which implies:

$$\lambda_o^2 + 4\lambda_o + 2 = 0 \qquad (1.6)$$

The solution of above equation λ_O = 0.5857 or 58%, that is the throughput of a tagged output port being analyzed. This concept was first presented by Karol, Hlychy, and Morgan from AT&T [7]. The detailed derivation of equation (1.1) through (1.6) is provided in Appendix 2 and 3. This analysis shows that unconditional use of input queuing results in 58% throughput due to the HOL blocking. In order to avoid the HOL blocking, we can drop off contention losers from the queues rather than keep them waiting at the HOL. Even so, the throughput is upper bounded at 63% due to the fact not all output ports are fully utilized.

Probability of a cell destined to a tagged output $= \frac{1}{N} \times \lambda_i$

which implies:

Throughput of the switch $= \lambda_o = 1-(1-\frac{\lambda_i}{N})^N$

Take $\lambda_i = 1$ for maximum throughput analysis,

$$\lambda_o = 1-(1-\frac{1}{N})^N \qquad (1.7)$$

if N=1 (1x1 switch), λ_O=1 or 100% throughput as expected. It is easy to see because there is no HOL blocking and no contention in one input and one output switch so 100% throughput is achieved. For N=2, throughput λ_O is .75 or 75% and for N=8, throughput is decreased to .6563 or 65%. For large N, equation (1.7) becomes:

$$\lambda_o = 1-e^{-1} = 63\%$$

which is the upper bound for large N with input queuing.

The delay analysis will not be discussed in this section since it is well known that

the delay at an input queue follows the M/D/1 queuing discipline. From the Pollaczek-Khinchin (P-K) formula [1], the mean delay can be shown as:

$$Mean\ Delay = \overline{D} = \frac{\lambda_i}{2(1-\lambda_i)} \qquad (1.8)$$

for service rate $\mu = 1$ time unit

4.2) ATM Switch with Output Queuing

The figure 9 depicts a switch with output queuing mechanism. All cells are routed to their respective output ports and stored there for services. There is no input queuing and consequently HOL blocking is avoided. Consider the same situation encountered in figure 8, where both Input 1 and Input N have cells destined to the same Output 3. These two cells with the same destination address Output 3 are routed over to Output 3 and stored there for services. Therefore, the second cell in Input N is able to go to Output 1, and higher throughput than input queuing shown in figure 8 is achieved.

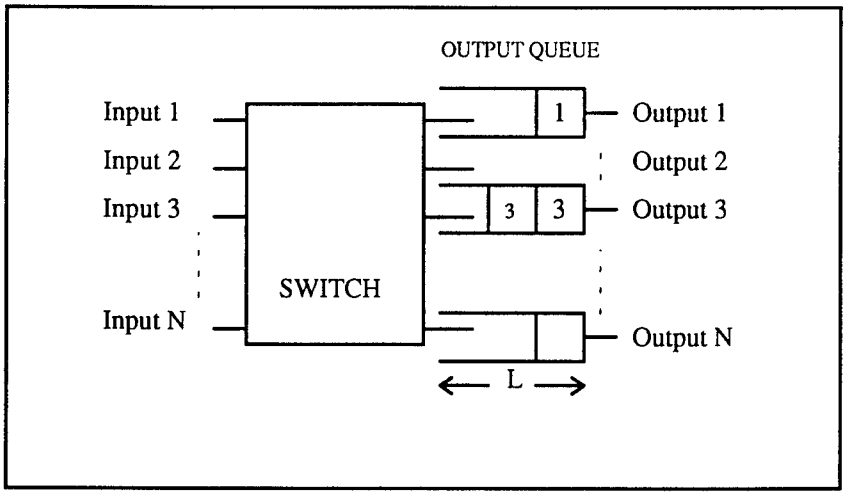


Figure 9. Switch with Output Queues

The analysis for throughput of a switch with output queuing technique is provided below:

P_K : Probability of K cells destined to a tagged output port

L : Queue size of a tagged output

$$P_K = \frac{N!}{K!(N-K)!}\left(\frac{\lambda_i}{N}\right)^K\left(1-\frac{\lambda_i}{N}\right)^{N-K} \qquad (1.9)$$

for large N, above equation can be simplified as:

$$P_K \approx \frac{\lambda_i}{K!}e^{-1}$$

Take $\lambda_i = 1$ for maximum throughput analysis implies:

$$P_K = \frac{e^{-1}}{K!}$$

If K is less then or equal to the output queue size L, all K cells will be admitted to the output queue. However, if K is greater than L, then only L cells are admitted into the output queue and remaining L-K cells will be dropped.

$$\therefore \quad \lambda_o = \sum_{K=0}^{L-1} KP_K + L\sum_{K=L}^{N} P_K$$

Use $\sum_{K=0}^{N} P_K = 1$

$$\lambda_o = \sum_{K=0}^{L-1} KP_K + L\left(1-\sum_{K=0}^{L-1} P_K\right)$$

Substitute $P_K = \frac{e^{-1}}{K!}$

$$\lambda_o = L\left(1-\sum_{K=0}^{L-1}\frac{L-k}{K!}\right)e^{-1}$$

If queue length L=1 or no queuing at all,

λ_0=1-e^{-1}=63%

If $L \rightarrow \infty$ then

$$\lambda_o = L - L\sum_{K=0}^{L-1}\frac{e^{-1}}{K!} + \sum_{K=1}^{L-1}\frac{e^{-1}}{(K-1)}$$

$$\lambda_o = L - Lee^{-1} + ee^{-1} = 1$$

$$\lambda_o = 100\%$$

Use of output queuing achieves 100% throughput. In fact, the output queue gives optimal throughput-delay performance. Generally, the queue length is selected according to the steady state traffic loading to avoid cell loss. However, the long burst of data can overload the queue at some time instances in the ATM network, so cell loss is not completely avoidable in the output queuing technique. In military network, it is more important that cell loss be minimized. The delay at each queue follows the same M/D/1 queuing discipline, as in equation 1.8.

4.3) ATM Switch with Input and Output Queuing

The Air Force SSCN project, which objective is to create an ATM based high performance high speed network, uses input and output queuing to reduce the cell loss. It has been shown that use of input and output queuing can achieve throughput performance of 78% with output queue size L of 4, and 100% throughput is achieved as L approaches

to the infinity [6]. The throughput analysis for input and output queuing can be performed based on the techniques provided in section 4.1 and 4.2, and will not be discussed in this section. The delay analysis in input and output queuing is, however, different from that of section 4.1 and 4.2, and discussed below:

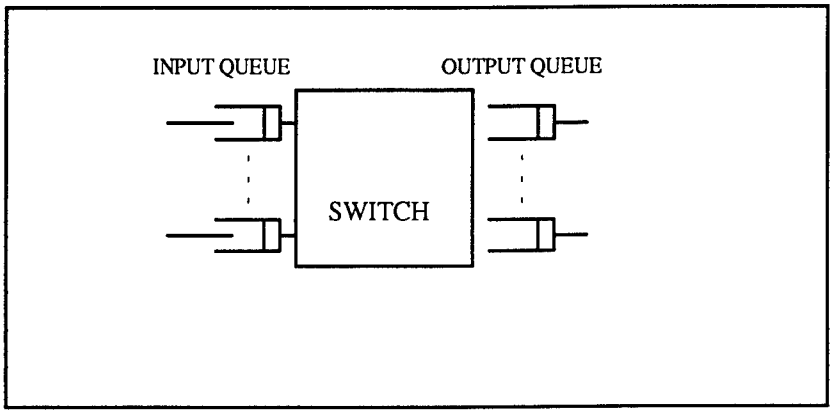


Figure 10. Switch with Input and Output Queues

The mean delay of input and output queuing consists three components:

1. Delay at input queue, that is from the time a cell enters the input queue to the time it reaches the HOL position
2. Delay at HOL, that is the time a cell waits at the HOL position before it is routed over to an output queue
3. Delay at output queue.

The combination of delay at the HOL in the input queue and the delay at output queue is governed by M/D/1 concept .

$$\overline{D}_{HOL+out\_q} = \frac{\lambda_i}{2(1-\lambda_i)} \qquad (1.10)$$

Delay due to the input queuing is governed by M/G/1 principle and can be shown as:

$$\overline{W_i}(\lambda_i) = \frac{\lambda \overline{T}^2(\lambda_i)}{2(1-\lambda_i \overline{T}(\lambda_i))} \qquad (1.11)$$

Therefore, the total average delay in input and output queuing is combination of equation (1.10) and (1.11).

$$\overline{D} = \overline{D}_{HOL+out\_q} + \overline{W_i} \qquad (1.12)$$

The detailed derivation of (1.10) through (1.12) can be found in [6]. Formulas for calculation of first moment of T is provided in Appendix 4. This input and output queuing scheme offers an alternative congestion control with high throughput achievement and smaller cell loss with the price of higher mean delay. It is critical, especially in tactical communications, that cell loss shall be minimized in order to achieve the mission. In next section, simulation models for output queuing and input and output queuing are discussed.

5) Simulation Models

Two switch node simulation models are developed in order to compare and evaluate the output queuing technique and the input and output queuing technique. A Switch with output queuing and a switch with both input and output queuing are

designed. Each simulation model has two versions: one with ideal[1] traffic source and one with bursty source, are designed. All simulation models are designed with object oriented approach with high modularity for future enhancement and improvement. Two bursty traffic generator models, which characterize the traffic patterns expected to be encountered in ATM network are discussed in section 5.1. Then, switch node model with two different queuing techniques are presented in section 5.3 and section 5.4.

5.1) Traffic Generator in ATM Networks

The poisson process is traditionally used as a probability distribution for packet interarrival time in the packet switching network. However, high speed transmission, high speed switching, and expected large data files transfer make poisson process invalid for characterizing the arrival process in future high performance BISDN network. It is expected that MMPP (Markov Modulated Poisson Process) is a good approximate random traffic model for ATM network [4]. However, it is generally agreed that the arriving traffic can be modeled with two-state Markov process: active state in which traffic will be generated with pertaining random parameters, and silent state in which no traffic will be generated. The 2-state model is used to generate bursty traffic in section 5.1.1 and 5.1.2, which is an essential component in queuing analysis for ATM switch simulation model. Most of the commercially available simulation tools do not supports bursty traffic generation capability so development of traffic generator software code is critical. In addition to the bursty traffic, the constant bit rate (CBR) normally used in video signal coding, and low bit rate and higher bit error rate (LBHER) are expected in

1. OPNET supports some well known traffic arrival patterns in its *ideal traffic generator* module. User can simply pull down the menu and click on the desired pattern such as CBR or Poisson, etc.

tactical ATM networks. These two types of traffic can be generated by OPNET built-in library and will not be discussed in this section.

5.1.1) Bursty Traffic Generator Mode 1

There are two operational modes, active mode and silent mode as shown below.

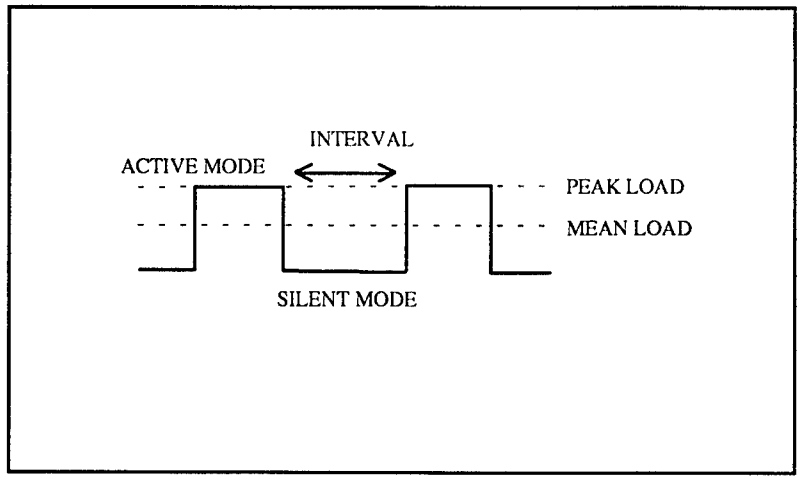


Figure 11. Two-state Traffic Generation Function

Bernouli probability is defined as:

$P\{x=k\} = 1-p$ for k=0

p for k=1

0 Otherwise

and Geometric probability function is defined as:

$P\{x=k\} = (1-p)^k p$ for k=0,1,....

$= 0$ Otherwise

Two operational modes are selected according to the Bernoulli process with user defined probability of success p, and the duration of each mode is assigned according to the Geometric probability with user defined probability p. This traffic generator model does not perfectly represent the characteristic of integrated traffic in BISDN, however, it accurately characterizes the randomness of arrival process and the distribution. This traffic generation technique, also known as two states on/off model is widely accepted in BISDN simulation. It has the capability to generate bursty traffic, which is not available on the current version of simulation tool (OPNET). The control parameters in this traffic generator models are: Peak Service Rate, which defines the maximum number of bit per second that can be generated by the generator; % of Peak Rate, which is the average offered load in terms of Peak Service Rate; Burst Length, which is the maximum number of packet or cell that can be transmitted with Peak Service Rate in one time slot. The figure 12 shows the bursty traffic outputs of the Model 1 with different control parameters.

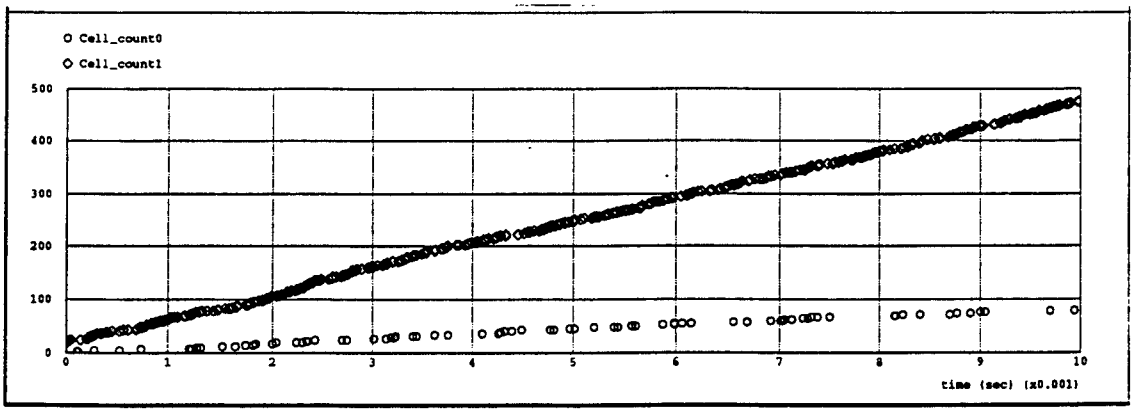


Figure 12. Different burst lengths generated by the same Traffic Generator Model 1. Cell_count0 has shorter burst length than Cell_count1

5.1.2) Bursty Traffic Generator Model 2

The Model 2 is similar to the Model 1 described in section 5.1.1. The burst length is governed by the Gaussian distribution with user defined parameters. This model allows user to define Probability of Success used in Bernoulli process for selecting operational modes; Average Mean Rate and Variance, which are variables in the Gaussian function; and Waiting Time, that characterizes the duration of silent mode. A simulation model named *sh4s_net*[1] is designed to demonstrate the traffic patterns generated by Model 2. The figure 13 shows the bursty traffic outputs of the Model 2 with different control parameters.

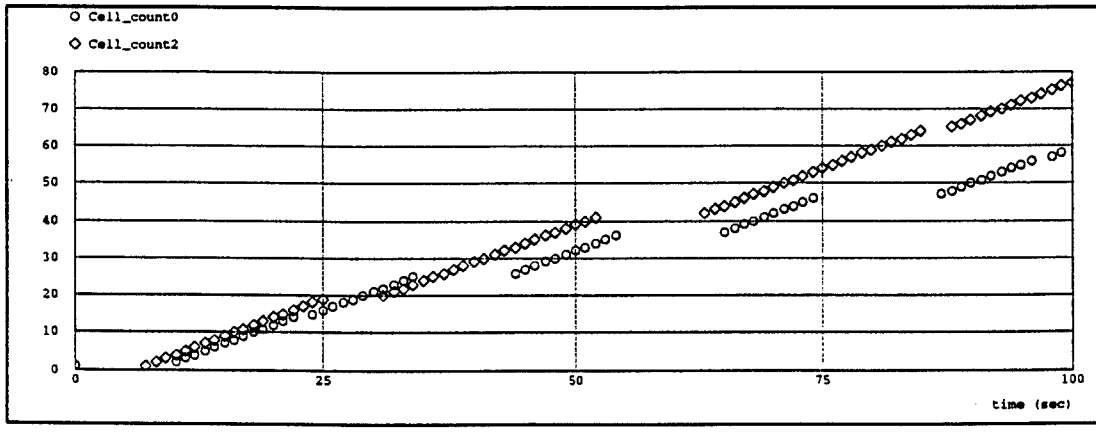


Figure 13. Different burst lengths generated by the same Traffic Generator Model 2. Cell_count0 has shorter burst length than Cell_count2

1. *sh4s_net* is a simulation model designed in OPNET to demonstrate the bursty traffic generated by Model2. Contact RL for the source code.

5.3) Switch with Output Queuing (SOQ)

The switch with output queuing (SOQ) model is developed to conduct performance analysis of output queuing. The concentration is given on queuing process and traffic flow rather than ATM switch architecture. As shown in Figure 15, this model comprises three components: switch node, output node, and sink node. The switch node contains traffic sources and an ATM switch. Two types of traffic sources, ideal source and bursty source, are considered in this simulation model. The ideal traffic source, which generates well known traffic patterns supported by OPNET built-in function. The bursty source class contains two bursty traffic generators described in 5.1.1 and 5.1.2. The offered load at all output port is statistically i.i.d (Independent and Identical Distribution) process so only one output port is required to analyze for the parameters of interest. Therefore, for the sake of memory and computation time complexity, the analysis is performed on one output port only. Nevertheless, expansion of input port and output port number can be done easily in OPNET simulation tool.

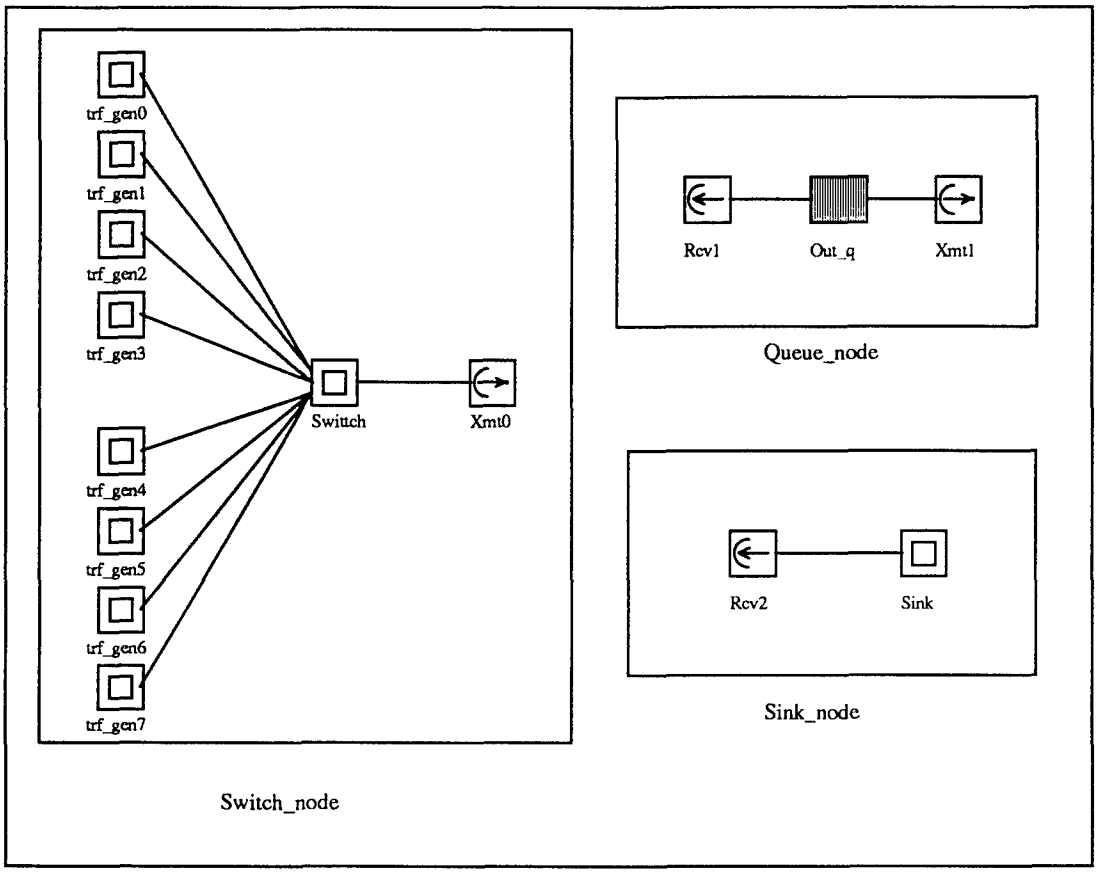


Figure 14. Node Modules for Output Queuing Scheme

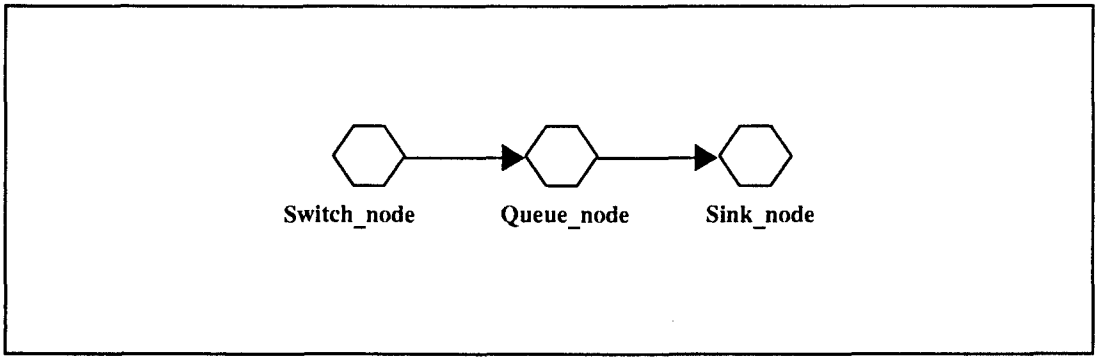


Figure 15. Simulation Model with Three Node Modules Shown in figure 14

As shown in above figures, the output queue is the main part of the output port. It serves cells with FIFO (First In First Out) discipline with user defined *service rate* and

the *queue size* (queue length). The sink node destroy the cells emerge from the output queue. It is an necessary component in this simulation tool in order to release the memory allocated for processing cells. Transmitters (xmt) and receivers (rcv) serve as interface between nodes.

Eight traffic sources generate ATM cells according to user selected characteristics such as constant bit rate (CBR) or bursty traffic. Then, these cells are routed to their respective output port by the ATM switch. In the worst case scenario, all arriving cells at the switch request for the same destination port so an output port can expect N cells at most in one time slot for a NxN switch. The switch in figure 15 is designed to address such worst case situation; it has only one output and it can route 8 cells in one time slot to demonstrate the worst case traffic loading at a particular output queue. If the queue is not full, arriving cells will be stored in output queue for service, and if the queue reaches its capacity, arriving cells will be discarded . The functions of output queue are controlled by the software code, which is embedded in the process model. The process model is an OPNET module that contains finite state machine to control the operational function of the node model. The sink node simply terminates cells upon arrival. This node level model can be expanded to network level model by removing sink node and connect the output port of the output queue to an input port of the second switch through appropriate transmitters and receivers.

The design provided in figure 15 is controlled by software code and very flexible. The object oriented nature of this simulation model allows user to modify and enhance any parts without having to redesign the whole simulation model. For example, user can change queuing discipline by just recreating the process models for the queue without

changing any other parts of the model. All process model is created with the C language in the UNIX environment. The output port throughput, cell loss, queue loading, and mean delay are analyzed in SOQ model.

5.4) Switch with Input and Output Queuing (SIOQ)

The switch with input and output queuing (SIOQ) model contains three nodes and a feedback path for backpressure signaling. Figure 17 depicts the SIOQ with ideal traffic source. As shown in figure 16, switch node contains a back pressure monitor device (bk_moni), and input queues, in addition to traffic generators (trf) and an ATM switch. Traffic generators and switch are same as those of SOQ model. The bk_moni monitors the *back pressure signal* which is issued by the output queue upon congestion, and *release signal*, which is generated when the output queue recovers some buffer space for arriving cells. In order to avoid the HOL blocking, input queues do not buffer any cells until the bk_moni receives a backpressure signal from the output queue which indicates that no more buffer space is available. By this way, cells are buffered at output queue first to maximize the throughput, and buffered at input queues instead of discarding them when output queue has no more buffer space available. The finite state machine provided in Appendix 5 describes the queuing discipline for the input queue.

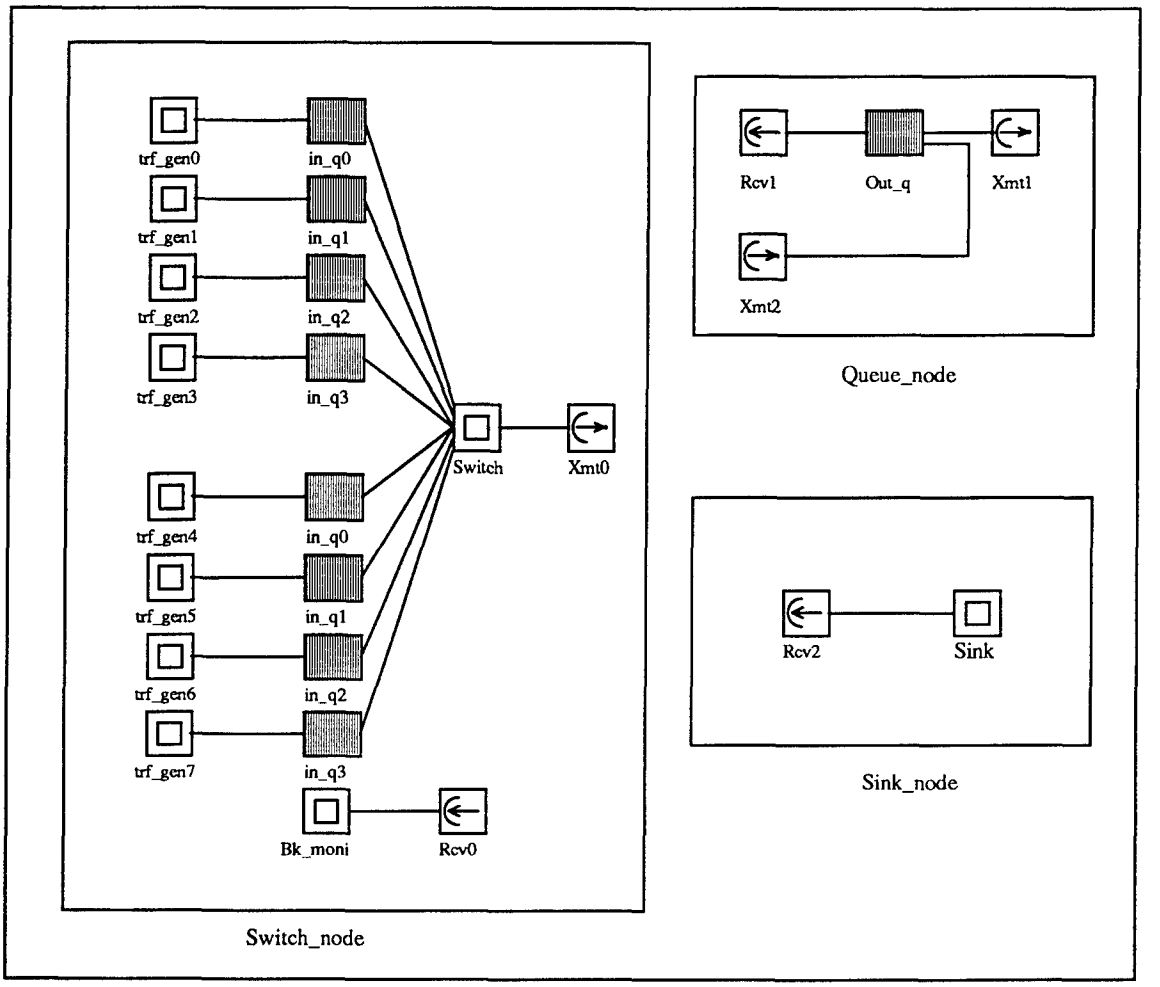


Figure 16. Node Modules for Input and Output Queuing Scheme

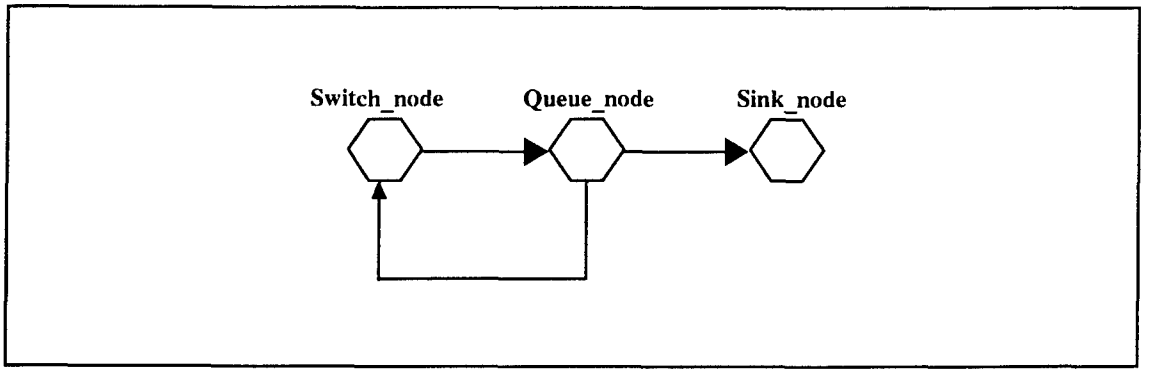


Figure 17. Simulation Model with Three Node Modules Shown in figure 18

Generally, a backpressure signal is generated when number of cells in output queue reaches the queue capacity. But, the backpressure signal takes at least one time slot to propagate back to the input side of the switch to inform its status. By the time input queues realize that output queue is full, there is already some cells on the way to the output queue that will be discarded upon arrival. Therefore, backpressure signal should be generated before output queue reaches its limit to accommodate cells being routed when it propagates back to the input side of the switch. Hence, threshold for generating backpressure signal is selected as:

$$Threshold = queuen\_size - N$$

where N is the number of input port.

Selection of threshold according to above rule seems to prevent the cell loss at output queue. However, our simulation work shows that some oscillation effect in queue length can be resulted, if single threshold is used. Consider a following case: upon reaching *queue_size -N*, output queue informs input queues with the backpressure signal about congestion. Input queues start buffering up cells when backpressure is received. Suppose, at the same time, output queue serves a cell so queue size at next time slot will be *q_size - N - 1*, which is below the threshold so input queues resume sending more cells to output queue again. This causes the output queue size exceeds the threshold and backpressure signal is sent back one more time to the input side of the switch. Upon receiving backpressure signal, Input queues start buffering up cells again and the whole process is repeated. This causes the output queue length oscillates around the threshold. The result of simulation shows that two thresholds should be used to prevent this

oscillating effect. This concept is similar to the Schmitt Trigger concept frequently used in digital electronics. One threshold is used to trigger the backpressure and the other one is used to generate the release signal for resuming the cell transmission from the input queue.

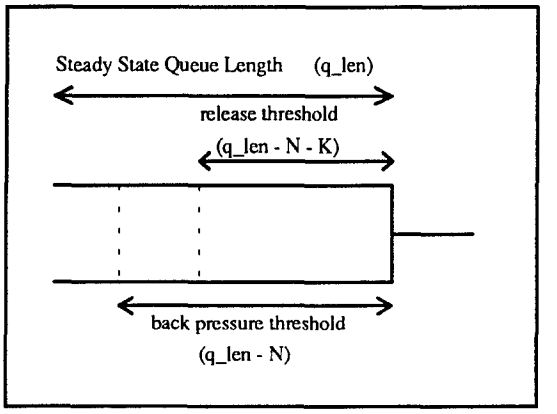


Figure 18. A Queue with backpressure generating threshold at *q_len - N* and congestion release signal threshold at *q_len - N - K* where *K* is greater than *N* .

6) Simulation Results

The figure 19 and 20 are obtained from simulation of figure 17 on SUN Sparc II machine. The maximum offered load to the output queue is 155Mbps x 8 inputs that is 1240 Mbps, and the output queue service rate is selected to be 775 Mbps (775Mbps<1240Mbps) to demonstrate the traffic loading, cell loss, back pressuring, and mean delay. The Simulation parameters and associated values are:

interarrival rate at input queue	= 2.735 μs (λ=155 Mbps or 365566 cells)
input queue service rate	= 155 Mbps
output queue service rate	= 775 Mbps
input queue size	= infinity
output queue size	= 100 cells
back pressure threshold	= 70 cells
release threshold	= 50 cells
sim_time	= 10 ms
real_time	= 460.727 sec

It should be noted that interarrival time is 2.735 μs because 155 Mbps (OC3 speed) or 365566 cells per second is selected as arrival rate.

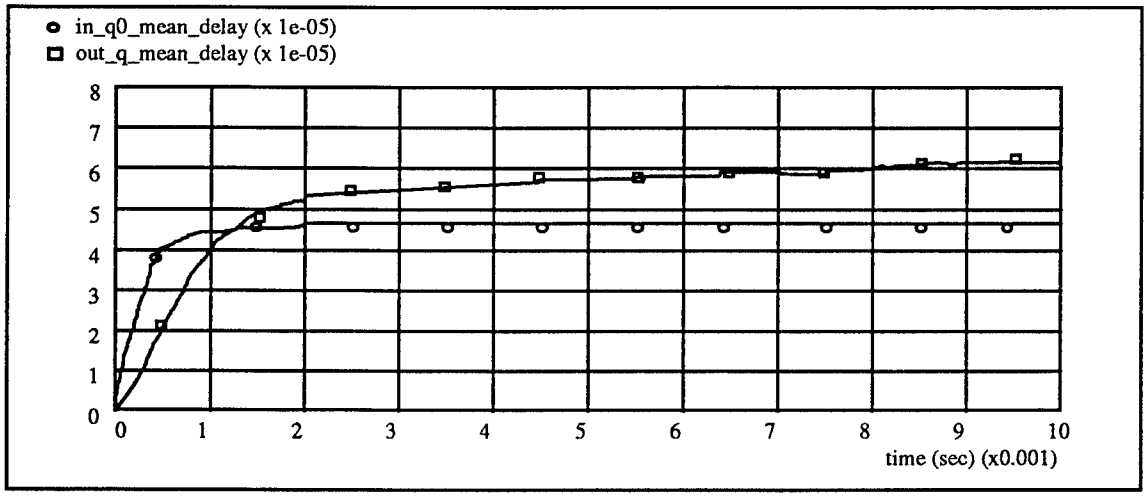


Figure 19. The mean delay in input queue and output queue for the output queue service rate of 775 Mbps

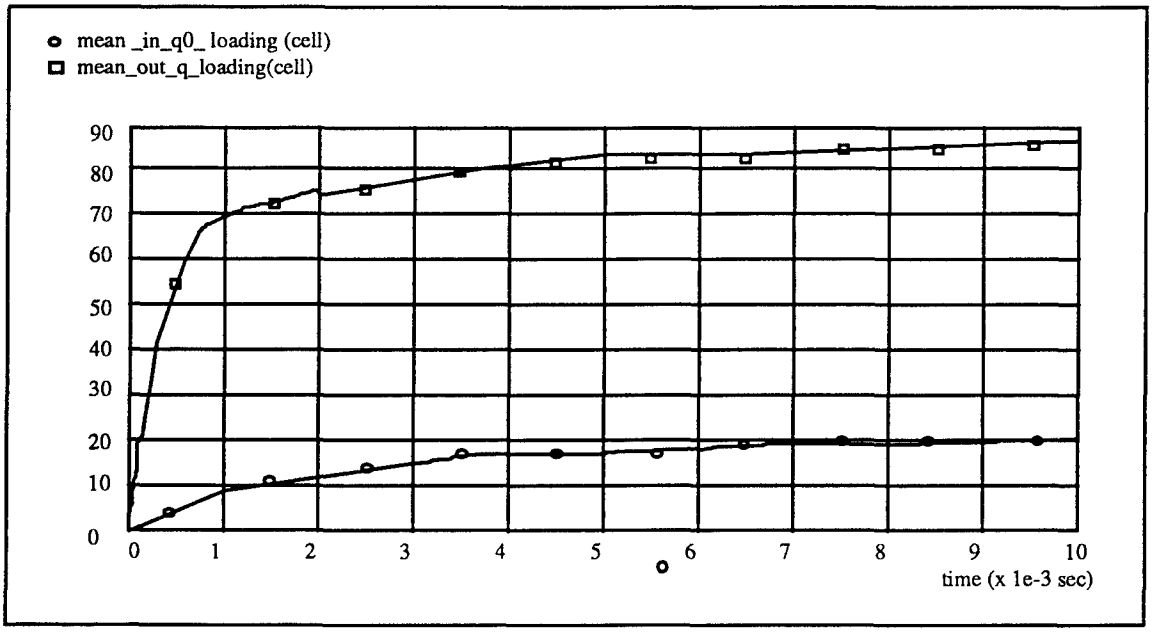


Figure 20. Average traffic loading in input queue and output queue for the output queue service rate of 775Mbps

Figure 19 shows that mean delay at input queue is greater than the mean delay at output queue due to the infinite length of input queue used in this simulation. Figure 20 depicts the traffic loading in both input queue and output queue in number of packets (cells). As expected, the output queue loading approach to 70 cells rapidly because the service rate 775 M bps is much smaller than offered load 1240 Mbps. When queue loading reaches 70 cells, the back pressure threshold, input queues are brought into action so the output queue length is stabilize around 86 cells. In this simulation model, input queues take two time slots (appendix 6) to realize that the output queue is congested. Within these two time slots, 16 cells can emerge from the input queues, therefore, 86 cells (70 cells + 16 cells) are expected in the output queue.

Then, output queue service rate is changed to 1550 Mbps, that is faster than the offered load of 1240 Mbps, to demonstrated the queue loading and average delay value.

The simulation parameters are provided below:

interarrival rate at input queue	= 2.735 μs (λ=155 Mbps or 365566 cells)
input queue service rate	= 155 Mbps
output queue service rate	= 1550 Mbps
input queue size	= infinity
output queue size	= 100 cells
backpressure threshold	= 70 cells
release threshold	= 50 cells
sim_time	= 10 ms
real_time	= 430.689 sec

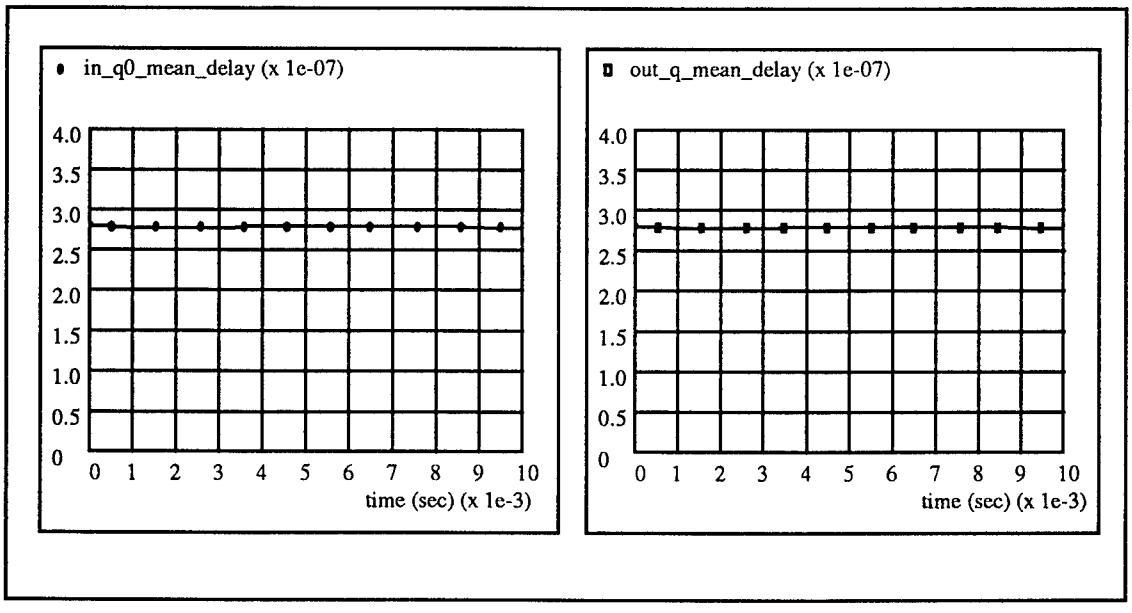


Figure 21. The mean delay in input queue and output queue for the output queue service rate of 1550 Mbps

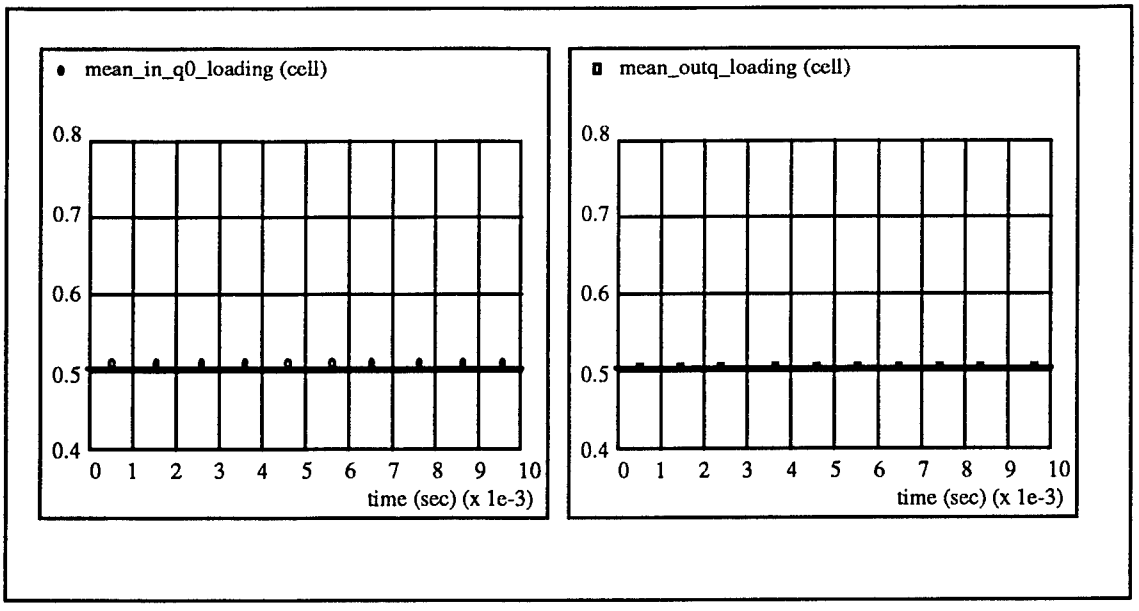


Figure 22. Average traffic loading in input queue and output queue for the output queue service rate of 1550Mbps

Figure 21. shows that mean delay at output queue service rate of 1550 Mbps is 27 μs which is much lower than 61 μs obtained with output queue service rate of 775 Mbps. Figure 22 shows that the average queue size is less than 1 cell in both input queue and output queue, as expected. The queue loading is always less than 1 cell since the service rate is higher than the offer load at the output queue, .

The above two simulations are intended for demonstration and verification on how accurate the simulation models response to different traffic loading and different service rates.

7) Recommendation for Future Research

The congestion control in a large scale theater network will be more difficult than

in a testbed network with small number of node. There are number of technical issues still yet to be addressed on large scale theater network. Even the threshold selection for back pressure signal requires extensive simulations under various scenarios. In addition to solving the congestion when it occurs, the preventive action is necessary in the ATM network, and must be capitalized. In commercial ATM network, the admission control is realized by traffic shaping, traffic regulating, and traffic enforcing functions. Admission control functional requirements shall be incorporated into the current congestion control mechanism to support the large scale theater level tactical network. Some important related research topics are Leaky Bucket Scheme from AT&T Bell Lab, Sequencer Algorithm from Bellcore and Virtual Clock Concepts from MIT. We all agree that effective preventive care is more important than the cure. Without effective preventive mechanism, the network will be disabled under worst case traffic loading. Two specific recommendations are provided below.

Recommendation 1:

The back pressure congestion control with input and output queuing shall be enhanced to further reduce the cell loss. In military network, traffic loading on some particular links at some time instance can be extremely heavy due to some loss of communications links and nodes. Use of input and output queuing can not guarantee zero cell loss. Instead of buffering cells at a single congested node, the distribution of queuing is suggested. Figure 23 shows how traffic can be buffered in distributed manner.

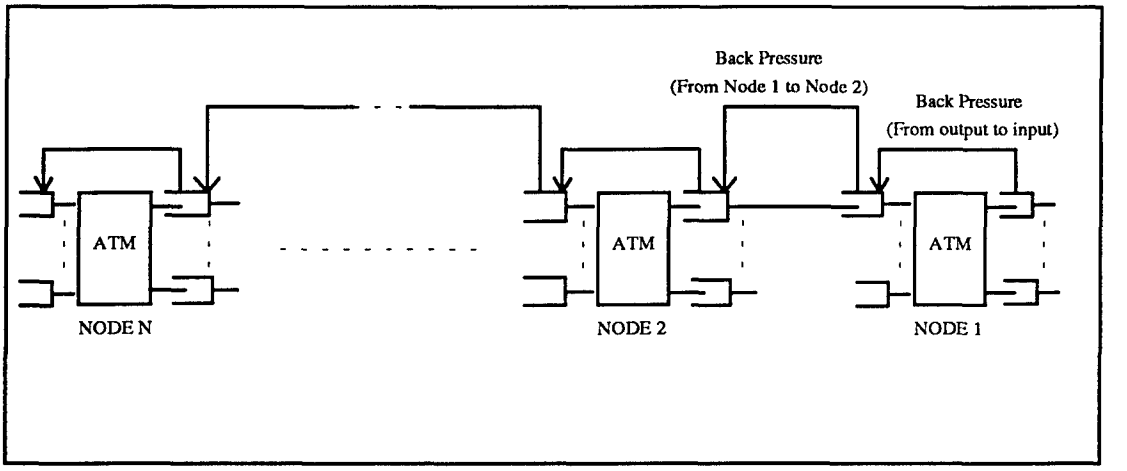
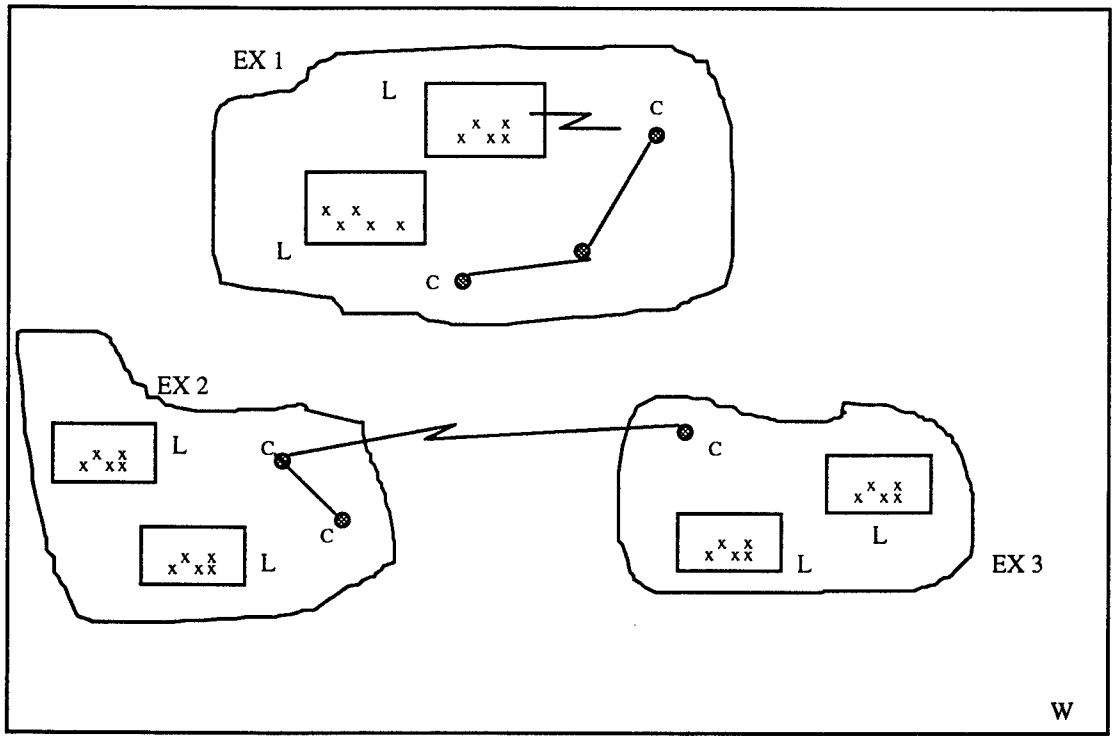


Figure 23. Node Level Backpressure Technique

Above figure shows how back pressure signal is sent back from a congested node to the source node. Let assume the Node N is the original source node and the Node 1 is a congested node. According to the scheme discussed in previous section, output queue of the Node 1 sends the backpressure signal (congestion acknowledge signal) back to the input side of the switch so input queue start buffering the traffic upon receiving of back pressure signal . If congestion continues and the traffic loading reaches the threshold, GTE proposed scheme start dropping low priority cells. But in Figure 23, the congested input queue of the Node 1 generates a back pressure signal and sends it back to the output queue of the predecessor node, Node 2, instead of discarding any cells. Output queue of Node 2 starts buffering traffic upon receiving of the back pressure signal from the Node 1. If the congestion continues so the output queue of Node 2 reaches its threshold, then the back pressure is again sent back to input queue of Node 2. By this way, back pressure is sent back to the source node, node by node, if the congestion continues. The traffic is, therefore, distributed among all queues along the line between the source node (Node N) and the congested node (Node 1). The important technical problem in this scheme is, once again, queue length selection and back pressure threshold

selection. Threshold shall be selected in the way that the queue still can accommodate the cells being transmit during the time backpressure propagates back to the queue before the congested queue. This task is not a simple one since communications support areas in tactical environment ranged from few ten miles to over thousands of miles.

Consider this technique in communications support areas defined in EDMUNDS (Evaluation and Development of Multimedia Networks in Dynamic Stress) project. The EDMUNDS project defines three support areas: Localized Area (30 miles x 100 miles), Extended Area (150 miles x 100 miles), and Wide Area (1000 miles x 1000 miles).



● ATM NODE
L: Localized Area (Below Echelon)
EX: Extened Area (Echelon)
W: Wide Area (Theater and Beyond)
C: Command Post
X: Fighting Unit

Figure 24. Wide Area Network (WAN) with ATM switches deployed in Extended Areas

In figure 24, the high speed communication link between command posts in extended areas is optical fiber media. The fiber is also used between extended areas in wide area together with satellite link. Consider an extended area for our analysis. The maximum possible distance between two nodes is 100 miles or 167 kilo meters.

The velocity of the light = $3x10^5$ Km.

Therefore, the maximum number of cells that can arrive at congested node when the back pressure signal is propagating back to the source node at the other end of the extended area can be approximated as:

$$\frac{167\ Km}{3\times10^5\ Km\times2.8\times10^{-6}} \approx 200$$

where $2.8x10^{-6}$second is the interarrival time of an ATM cell in OC3 speed. The above result indicates that as many as 200 cells can arrive to the congested node by the time the back pressure signal reaches the source node located 100 miles away (167 Kilo meters). However, buffering at intermediate points between the congested node and source node can reduce the buffer space requirement.

The distance between the Node 1 and the Node N in extended area can be 167 Kilo meters, and the Node 1 requires buffer space of 200 cells if it sends the backpressure signal directly back to the Node N. But, as shown in figure 23, if Node 1 sends the backpressure signal to its predecessor node, the buffer space requirement will be reduced to few ten cells for few ten kilo meter distance. Then Node 2 can backpressure the Node 3, which is a predecessor node of Node 2 and may be only few Kilo meters away.

This techniques can reduce the buffer space or queue capacity requirement at single node to the order of K for sending back pressure through K nodes which are located between the source node and the congested node, and distribute the queue capacity requirement among all nodes located between the source node and the congested node.

It should be noted that this recommendation is based on the initial study, and should not be considered as a conclusive statement. Also, this technique will be successful if and only if the predecessor node is no more than few ten Kilo meters away.

Recommendation 2

The admission control mechanism shall be incorporated into current simulation model to analyze the traffic loading under different traffic shaping algorithms. Figure 25 illustrates the complete simulation model with recommendation 1 and recommendation 2.

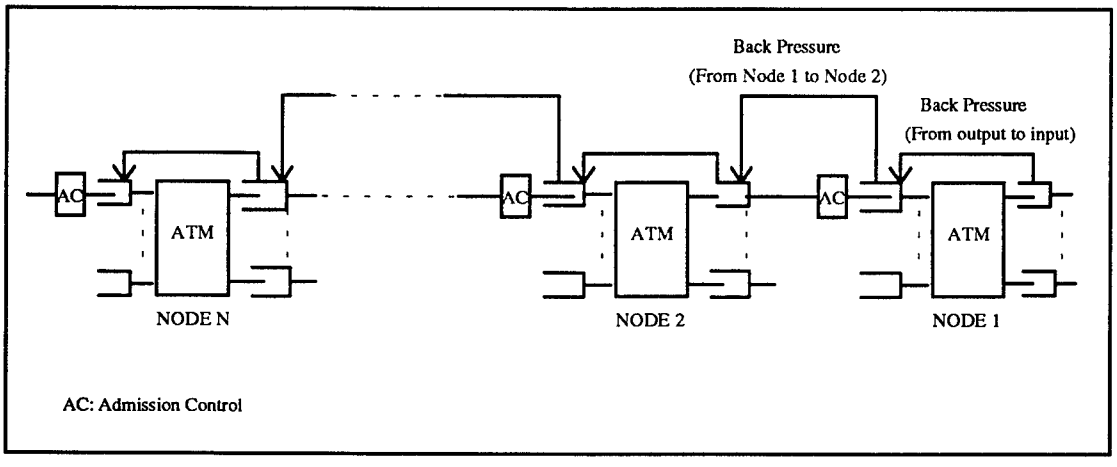


Figure 25 Node Level Backpressure Technique with Admission Control Mechanism

As shown in above figure, admission control (AC) mechanism is placed before the input queue. This admission control mechanism regulates the arrival and ensures that all virtual channels conform with the call setup parameters such as mean burst length, mean arrival rate, peak burst length, and peak arrival rate. This action is essential to prevent misbehaving users from occupying all the bandwidth, preserve the fair treatment among users with same priority level, and provide the better control capability for traffic loading and traffic flow in the buffers. Some well known AC algorithms such as Leaky Bucket scheme, Sequencer Algorithm, and Virtual Clock concept shall be incorporated into the current existing simulation model to study the responses in tactical environment.

8. Conclusion:

The result of this study indicates that use of input and output queuing reduces the cell loss probability with the price of higher mean delay. It is important to note that unconditional use of input queuing reduces the throughput due to the HOL blocking. The threshold selection for triggering backpressure signal requires more simulations under different scenarios; it is strongly suggested that two thresholds shall be used to trigger the backpressure signal. The simulation part of this research concentrated on the single node model due to the availability of time, resources, and limited computer support from the University. Nevertheless, this research work supports the RL with the capability to analyze the traffic loading, traffic flow, cell loss, mean delay under different arrival patterns, and laydown the ground work for the network level simulation and end-to-end performance analyses essential to the execution of both the "Global Grid" and "Artemis" Advanced Technology Demonstrations.

References:

1. D. Bertsekas and R. Gallager, "Data Network", Second Edition, Prentice Hall, 1992

2. GTE Government Systems, "Secure Survivable Communications Network", Technical Proposal Vol. III, March 1992

3. B. W. Hoe, "Congestion Control for ATM Network in a Tactical Theater Environment", Research Proposal for AFOSR sponsors SREP, August 1992.

4. H. Heffes, "A Markov Modulated Characterization of Packetized Voice and Data Traffic and Related Statistical Multiplexer Performance", IEEE JSAC, vol. sac 4 no. 6, Sep. 1986

5. J. Hui, "Switching and Traffic Theory for Integrated Broadband Networks", Kluwer Academic Publishers, 1990

6. I. Iliadis and W. Denzel, "Performance of Packet Switches with Input and Output Queuing", SUPERCOM/ICC '90, pp. 747 - 753, 1990

7. M. J. Karol, M. G. Hluchyj, and S. P. Morgan, "Input versus output queuing on space-division switch," IEEE Transaction on Communications, vol. 35, no. 12, pp. 1347-1356, Dec. 1987.

8. S.C. Liew and T.T. Lee, "N log N Dual Shuffle-Exchange Network with Error-Correcting Routing", Bellcore Technical Paper, 1993

9. Y.S. Yeh, M.G. Hluchyj, and A.S. Acampora, "The Knockout Switch: A Simple, Modular Architecture for High Performance Packet Switching," Proceedings of International Switching Symposium '87, March 1987

APPENDIX 1:

Hot Spot Effect in Tactical Environment

Consider the tactical network shown below. The switch j has N inputs and N outputs. It is highly possible that one of the output port becomes a hot-spot (trouble spot) under war time scenario as illustrated in figure A.1 and figure A.2.

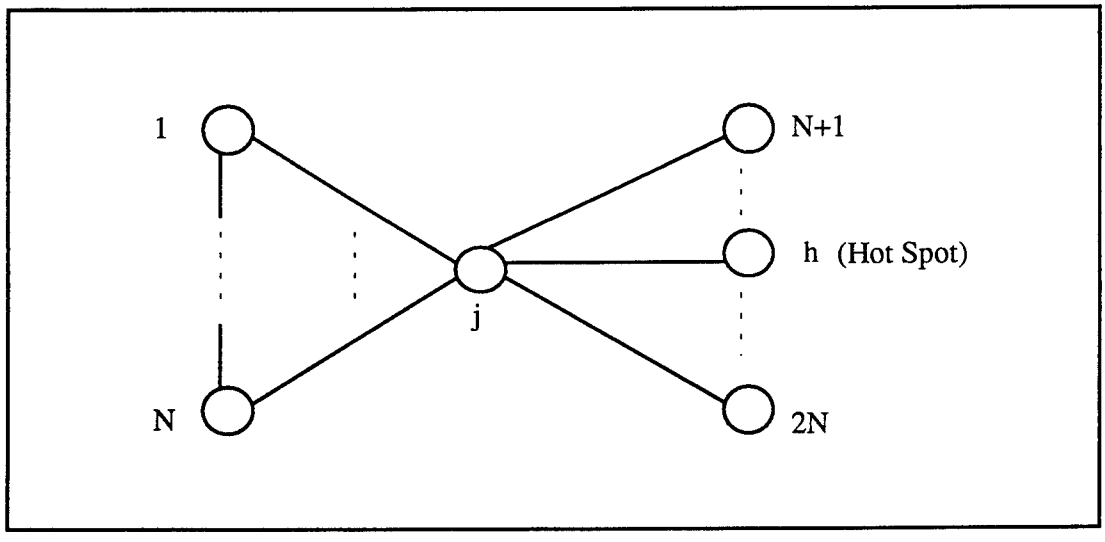


Figure A.1. A network with a hot-spot in node h

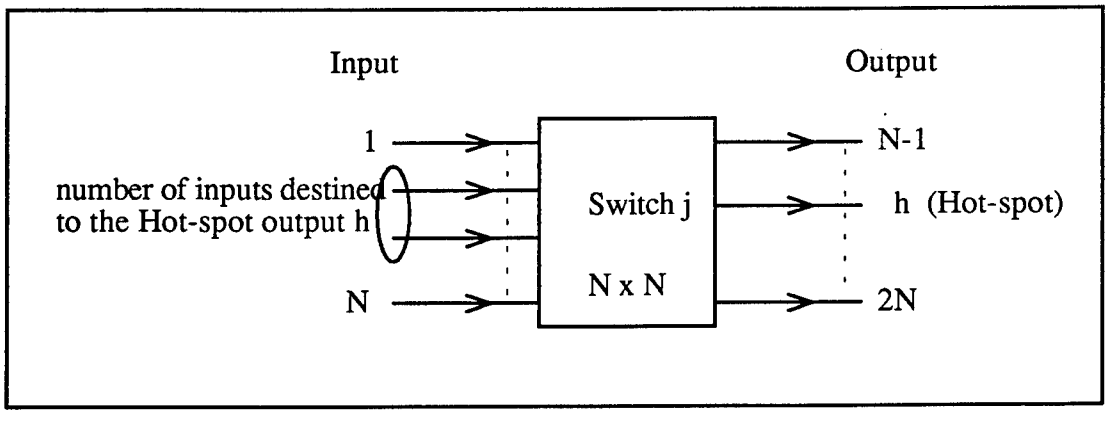


Figure A.2. Switch j from figure A.1 with hot-spot at output port h

(A.1)

The probability of K cells destined to the hot-spot can be analyzed as shown below.

Assume:

q : Probability of number of inputs destined to the hot-spot h

1-q : Probability of number of inputs destined uniformly and randomly to N outputs

therefore, the offered load at the input is:

$\lambda = q\lambda + (1-q)\lambda$

$$\Pr\{\text{a cell destined to the hot-spot } h\} = q\lambda + (1-q)\frac{\lambda}{N}$$

Probability of K cells destined to the hot-spot h is:

$$P_k\{\text{hot-spot}\} = \frac{N!}{K!(N-K)!}(q\lambda + (1-q)\frac{\lambda}{N})^K(1-q\lambda-(1-q)\frac{\lambda}{N})^{N-K} \qquad \text{(A.1)}$$

The above result for the probability of K cells destined to the hot-spot is not as simple as the probability of K cells destined to a particular output port shown in equation 1.9 of section 4.2, which is:

$$P_K = \frac{N!}{K!(N-K)!}(\frac{\lambda}{N})^K(1-\frac{\lambda}{N})^{N-K} \qquad \text{(1.9)}$$

The equation 1.9 is for well-behave environment such as commercial network, and its P_k depends on the arrival λ . The equation A.1 is more suitable for the network in a tactical environment; P_k {hot-spot}depends on the arrival λ and the q, which is the number of inputs destined to the hot-spot, and is difficult to predict.

(A.2)

APPENDIX 2:

Derivation of Equation 1.2 from Equation 1.1

From equation 1.1,

$$N_u = N - \sum_{j=1}^{N} [N_j - \in (N_j)] \qquad (1.1)$$

$$E[N_u] = N - \sum_{j=1}^{N} E[N_j - \in (N_j)]$$

$$E[N_u] = N - N\{E[N_j] - E[\in (N_j)]\}$$

$$\frac{E[N_u]}{N} = 1 - \{E[N_j] - E[\in (N_j)]\}$$

But, $E[N_U]\lambda_i = N\lambda_o$ (Flow Conservation Law)

$$\therefore \frac{\lambda_o}{\lambda_i} = 1 - \{E[N_j] - E[\in (N_j)]\}$$

$$\Rightarrow E[N_J] = 1 + \lambda_O \{\frac{E[\in (N_J)]}{\lambda_O} - \frac{1}{\lambda_I}\}$$

Substitute $E[\in (N_j)] = \lambda_o$

$$E[N_J] = 1 + \lambda_O \{1 - \frac{1}{\lambda_I}\} \qquad (1.2) \qquad \text{QED}$$

APPENDIX 3:

Derivation of Equation 1.4 from Equation 1.3

From equation 1.3,

$$N'j = N_j - \in(N_j) + A_j \qquad (1.3)$$

Squaring the equation 1.3 implies:

$$N_j'^2 = N_j^2 + \in^2(N_j) + A_j^2 - 2N_j \in(N_j) + 2N_j A_j - 2 \in(N_j) A_j$$

$$E[N_j'^2] = E[N_j^2] + E[\in(N_j)] + E[A_j^2] - 2E[N_j] + 2E[N_j]E[A_j] - 2E[\in(N_j)]E[A_j]$$

where $\in^2(N_j) = \in(N_j)$ and $N_j \in(N_j) = N_j$ because $\in(N_j)$ can be either "0" or "1" only.

Substitute $E[A_j] = E[\in(N_j)]$, and $E[N'^2_j] = E[N^2_j]$ in above equation gives:

$$E[A_j] + E[A_j^2] - 2E[N_j] + 2E[N_j]E[A_j] - 2E[A_j]E[A_j] = 0$$

$$E[N_j] = \frac{E[A_j] + E[A_j^2] - 2E^2[A_j]}{2(1 - E[A_j])}$$

$$E[N_j] = E[A_j] + \frac{E[A_j(1 - A_j)]}{2(1 - E[A_j])}$$

$$\Rightarrow E[N_j] = \lambda_o + \frac{\lambda_o^2}{2(1 - \lambda_o)}$$

because $E[A_j] = \lambda_o$, and $E[A_j(1 - A_j)] = \lambda_o^2$.

$$\therefore E[N_j] = \frac{\lambda_o(2 - \lambda_o)}{2(1 - \lambda_o)} \qquad (1.4)$$

QED.

(A.4)

APPENDIX 4:

Expression for $\overline{T}(\lambda_i)$ in equation (1.11)

From equation (1.11),

$$\overline{W_i}(\lambda_i) = \frac{\lambda \overline{T}^2(\lambda_i)}{2(1-\lambda_i \overline{T}(\lambda))}$$

where the service time at the input queue $\overline{T}(\lambda_i)$ is waiting time inside the queue $\overline{W}(\lambda_i)$ plus one time unit for its switching to the output port.

$\overline{T}(\lambda_i) = \overline{W}(\lambda_i) + 1$

Define the average queue length in cells as $\overline{Q}(\lambda_i)$.

Therefore, $\overline{W}(\lambda_i) = \frac{\overline{Q}(\lambda_i)}{\lambda_i}$.

APPENDIX 5:

The input queue enters to the arrival state (arrival) from the initial state (init) when a cell arrives at the queue. From arrival state, the input queue checks the "go_ok" flag which is initially set to "rele", which means the output queue is not congested. The input queue will go into the beginning of service state, "svc_start", if either the "go_ok" flag is in the "rele" mode, and there is a successful insertion of a new packet and the server is not busy (go_ok==rele && inset_ok && !server_busy) or there is no new arrival but the queue is not empty (!QUEUE_EMPTY). The "get_ok()" function is a conditional function that shows the status of the input queue. From the "svc_start" state, the input

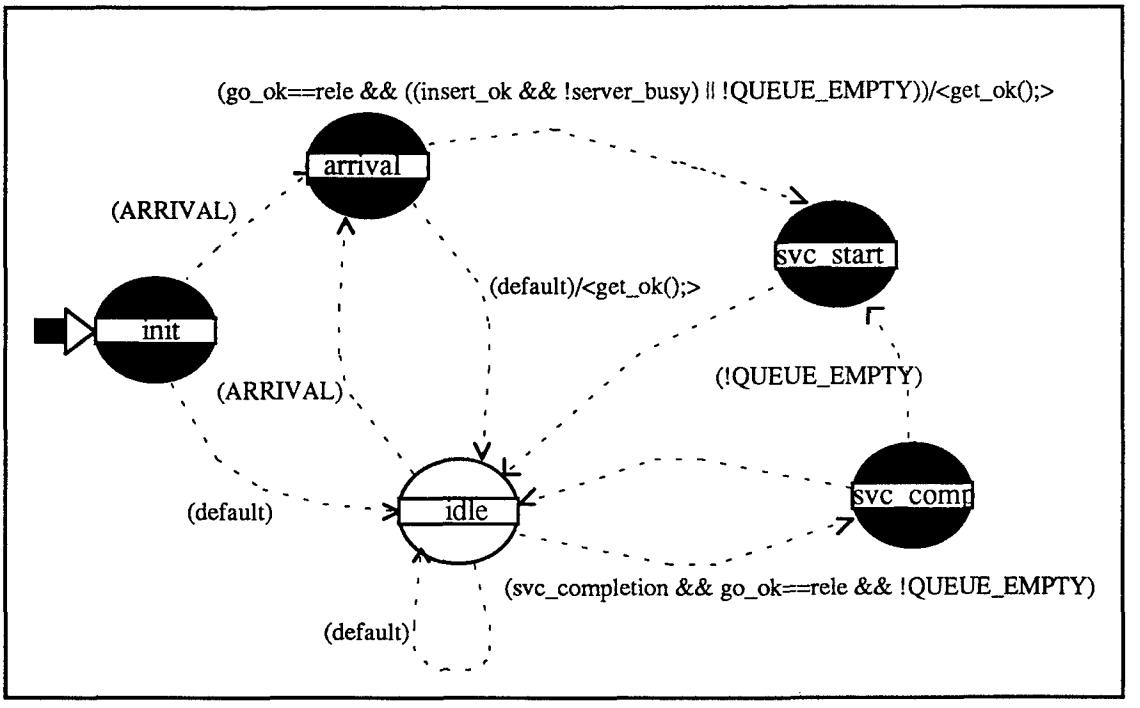


Figure A3. The State Machine for Input Queue in the SIOQ

(A.6)

queue enters the idle state upon completion of servicing a cell. Then, it moves to the "svc_comp" state if a service is completed for a cell and the output queue is not congested and the queue still have more cells to be served (svc_completion && go_ok==rele && !QUEUE_EMPTY). From the "svc_comp" state, the input queue will move back to "svc_start" state and serve another cell if the queue is not empty yet. The "idle" state is used as a transition state, and an idle state. All state diagrams are embedded in the source code delivered to the RL, and will not be described in this paper.

APPENDIX 6:

The backpressure signal takes 1 time slot to propagate back to the input side of the queue. After it reaches the bk_moni at the input side, it takes another time slot for input queues to realize that back pressure signal was sent. Using of bk_moni to detect the back pressure signal rather than sending back pressure signal directly back to input queues seems silly. However, it is a good practice to separate an unique function as a component, at least in simulation modeling point of view. It gives the flexibility in modifying or incorporating the existing function with future work.

AUTOMATED NATURAL LANGUAGE EVALUATORS - (ANLE)

Khosrow Kaikhah
Assistant Professor
Department of Computer Science

Southwest Texas State University
601 University Dr.
San Marcos, Texas 78666

Final Report for:
Research Initiation Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Boling Air Force Base, Washington, D.C.

and

Southwest Texas State University

December 1993

AUTOMATED NATURAL LANGUAGE EVALUATORS - (ANLE)

Khosrow Kaikhah
Assistant Professor
Department of Computer Science
Southwest Texas State University

Abstract

By the turn of the century, it is expected that most computer applications will include a natural language processing component. Both developers and consumers of NLP systems have expressed a genuine need for standard natural language system evaluators. Automated natural language evaluators appear to be the only logical solution to the overwhelming number of NLP systems that have been produced, are being produced, and will be produced.

The system developed here is based on the Benchmark Evaluation Tool [7] and is the first attempt to fully automate the evaluation process. This effort was accomplished in two phases. In phase one, we identified a subset of the Benchmark Evaluation Tool for each class of NLP systems. And in phase two, we designed and implemented a natural language generation system to generate non-causal semantically meaningful test sentences. The generation system can be queued for each class of NLP systems.

We followed an Object-Oriented Design (OOD) strategy. In this approach all concepts, including semantic and syntactic rules, are defined as objects. Each test sentence is generated as a chain of words satisfying a number of semantic, syntactic, pragmatic, and contextual constraints. The constraints imposed on the generation process increase dynamically while the sentence is being generated. This strategy guarantees semantic cohesiveness while maintaining syntactic integrity. In this approach, syntactic and semantic knowledge were utilized concurrently in word-objects. Each word-object is an independent knowledge source with local knowledge that can decide whether it can be a part of the sentence being generated, when called upon by the sentence-generator to join the chain.

Table of Contents

AUTOMATED NATURAL LANGUAGE EVALUATORS - (ANLE)

Khosrow Kaikhah

1.0 Introduction

The NLP community is rapidly growing, as a consequence, so is the number of NLP systems. However, evaluation of such systems has not received as much attention. In an effort to standardize the evaluation process, Calspan Corporation proposed and implemented the Benchmark Evaluation Tool for evaluating natural language processing systems [7]. The tool was designed to measure the linguistic capabilities of NLP systems regardless of the domain for which the NLP system was intended for.

The Benchmark Evaluation Tool was applied to PUNDIT during the summer of 1992 with the following conclusions [5]: A) The evaluation process must be customized for each class of natural language processing system, since NLP systems are designed for well-defined objectives and domains; and B) The evaluation process must be automated, since it is extremely long and time consuming. Generally, the NLP systems can be categorized into five classes: Database Management Systems, Command & Control Systems, Decision-Aiding Systems, Engineering Design Systems, and Diagnostic Systems. We attempted the automation process in two phases. In phase one, we identified an applicable subset of the Benchmark Evaluation Tool for each of the five NLP system classes; and in phase two, we designed and implemented an object-oriented system for generating *non-causal meaningful* sentences. The system is implemented in CLOS *(COMMON LISP Object System)*, an object-oriented extension of COMMON LISP.

This report will cover the object-oriented analysis, design, and implementation of the system. It will describe the objects that were used in generating a sentence and the way these objects interact to constrain the generation process. It highlights the major design and implementation issues and demonstrates the internal behavior of the system with two detailed examples. We conclude the report with sample sentences and a discussion of extension possibilities.

2.0 Discussion of the Problem

Both NLP developers and consumers have expressed a genuine need and desire for a standard evaluation procedure. NLP systems can be evaluated in several areas including: linguistic competence, end user issues such as reliability and likability, system development issues such as maintainability and portability, and intelligent behavior issues such as learning and cooperative dialogue. The underlying goal of the Benchmark Evaluation Tool was to create a product to test the linguistic capabilities of NLP systems, independent of the application under investigation. This feature is unique, in that, the tool is sensitive to each individual linguistic capability and not to each individual application.

The Benchmark Evaluation Tool [7] is composed of twelve independent sections which are designed to progressively test different linguistic features of an NLP system. The twelve sections are: 1) Basic Sentences, 2) Interrogative Sentences, 3) Noun Phrases, 4) Adverbials, 5) Verbs and Verb Phrases, 6) Quantifiers, 7) Comparatives, 8) Connectives, 9) Embedded Sentences, 10) Reference, 11) Ellipsis, 12) Semantics of Events. Each section is composed of a collection of evaluation procedures, each designed to test a single linguistic feature. These procedures include: an explanation of the feature being tested, a sentence template, example sentences, and criteria to use in evaluating the behavior of the NLP system.

After using the Benchmark Evaluation Tool to evaluate an NLP system, PUNDIT, we concluded that although the tool was extremely helpful in providing a guideline for testing the system, there were a number of clashes between the wide scope of the evaluation tool and the narrow application domain of the NLP system. Each class of NLP system possesses certain attributes that are unique. Each class has strengths and weaknesses which are directly associated with the goals and objectives of the system. Therefore, the evaluation procedure should place more emphasis on the class and objectives of the system and should be designed to test the applicable sentences to the system rather than the non-applicable sentences. For instance, if the NLP system is a Database Management System, the evaluation tool should place more emphasis on interrogative and basic sentences rather than ellipsis or quantifiers.

There are, naturally, sentences that are applicable to NLP systems and their domains. However, there are two completely distinct types of non-applicable sentences: A) non-applicable to the system, and B) non-applicable to the domain. Non-applicable to the system type sentences are those sentences where a meaningful sentence can not be formed with the suggested grammatical structure and available vocabulary. For instance, a Data Base Management system is not designed to handle a passive voice sentence. Non-applicable to the domain type sentences are those where a sentence can be constructed with the available vocabulary that can satisfy the suggested grammatical structure, but is not semantically meaningful. For instance, the sentence: 'Does the Atlanta to Atlanta flight have a stop in Atlanta?' should not be generated. Instead, the sentence: 'Does the Atlanta to Denver flight have a stop in Boston?' may be generated. The first phase of our project is designed to eliminate the non-applicable to the system sentences, while the second phase concentrates on the non-applicable to the domain sentences.

3.0 Methodology

A sentence at different levels is controlled by different forces. For instance, at the top level, a complete sentence is controlled by its main verb. The main verb determines the type, person, and in some cases the number of its subject and direct or indirect objects as well as their modifiers. Each phrase is also dominated by different forces. For instance, in a noun phrase, the noun determines the type and number of modifiers; in a prepositional phrase, the preposition is the driving force; and in a verb phrase, the verb is the dominating force. In our approach, a sentence is generated as a chain of words satisfying a number of semantic, syntactic, pragmatic, and contextual constraints concurrently. This chaining process ensures the compatibility of different components and results in a cohesive and unambiguous sentence.

Our goal was to design and develop a system that could generate semantically meaningful English sentences. The sentences were to be built according to a grammatical structure, or template, for a specified NLP system class, and were to be non-causal. In other words, there was no desired intention or content specified for the sentences. In our design we have utilized Object-Oriented Design (OOD) techniques [1] to develop a system where static objects of knowledge interact with dynamic elements of context in order to recursively build a sentence piece by piece. In this interactive mode, objects that have already been built are used to adjust the syntactic and semantic requirements of future sentence objects. We chose the object-oriented approach for the following two reasons: a) Knowledge can be divided into two general categories, static or functional. The object-oriented framework provides a convenient way to combine the two in the same data structure, an object; b) A large amount of data is used to acquire different types of knowledge. The uniform treatment of data and knowledge facilitates the maintenance of the system and promotes the reuse of system functionalities. This section describes, in detail, the design and implementation of our system.

3.1 Phase One

Although the idea of testing the sensitivity of individual linguistic capabilities of NLP systems rather than the sensitivity of the systems to individual applications is extremely attractive, it has nevertheless proved to be an ambitious task [5]. Most NLP systems are designed for well-defined domains and applications. Therefore, a general purpose

evaluation tool may not be suitable for all classes of NLP systems. This was evident from our prior experience. In order for an automated evaluator to be successful, the evaluation should be performed within the scope of the NLP system. Therefore, there should be several different automated evaluators each specialized for a different class of NLP system. Each automated evaluator would have syntactic, semantic, and pragmatic knowledge of only one class of NLP system and would generate appropriate test sentences. In an effort to accomplish this goal, we attempted to identify a subset of the Benchmark Evaluation Tool for each class of NLP systems.

The first phase of our work involved building tables, similar to Table I, for each of the twelve sections outlined in the Benchmark Evaluation Tool [7]. We evaluated each syntactic feature, proposed for testing, for its applicability to each of the following NLP system classes: Database Management Systems, Command & Control Systems, Decision-Aiding Systems, Engineering Design Systems, and Diagnostic Systems. If the feature seemed applicable, it was marked with an "X", otherwise it was left blank. NA was entered in the columns, if the feature was not applicable to any class. For example, every template, except Passive Voice, was selected as applicable from the Basic Sentences section for Database Management Systems. From this effort we were able to build a profile of applicable sentences for each system class to use in the generation of sentences. These profiles[1] were used to limit the template choices available to the selected NLP system and allow access to only a subset of all the grammatical structures. Therefore, if the class of NLP system being evaluated is Command & Control, only those features applicable to its class will be examined.

[1] The complete set of profiles are available upon request.

	Database Management Systems	Command & Control Systems	Decision-Aiding Systems	Engineering Design Systems	Diagnostic Systems
I. Basic Sentences					
1. Basic Sentence Types					
1.1 Declarative Sentences	X	X	X	X	X
1.2 Imperative Sentences	X	X	X	X	X
1.3 Interrogative Sentences	X	X	X		X
2. Simple Determiners					
2.1 The Indefinite Article	X	X	X	X	X
2.2 The Definite Article	X	X	X	X	X
3. Simple Noun Phrases					
3.1 Count Nouns	X	X	X	X	X
3.2 Proper nouns	X	X	X	X	X
3.3 Mass Nouns	X	X	X	X	X
4. Simple Verb Phrases					
4.1 Copular Verb Phrases	X		X	X	
4.2 VP Involving the Verb Do					
4.2.1 DO used in Interr.	X	X	X		X
4.2.2 The Emphatic DO	X	X	X	X	X
4.3 Transitivity					
4.3.1 Simple Transitive VP	X	X	X	X	X
4.3.2 Simple Intransitive VP	X		X	X	
4.3.3 Simple Ditransitive VP	X	X	X	X	X
4.4 Voice					
4.4.1 Active Voice	X	X	X	X	X
4.4.2 Passive Voice	NA	NA	NA	NA	NA

Table I: Section I - Basic Sentences

3.2 Phase Two

Based on our previous experience, a human evaluator tests an NLP system by inputting a number of random sentences and observing the system's response [5]. These random sentences are not constructed to express a particular intention, but rather are constructed to be within the scope of the NLP system in terms of grammar and vocabulary. Hence they are non-causal. Our goal was to automate the generation process of these non-causal sentences. Therefore, we concentrated on generating semantically meaningful and syntactically correct non-causal sentences and ignored intention.

3.2.1 System Overview

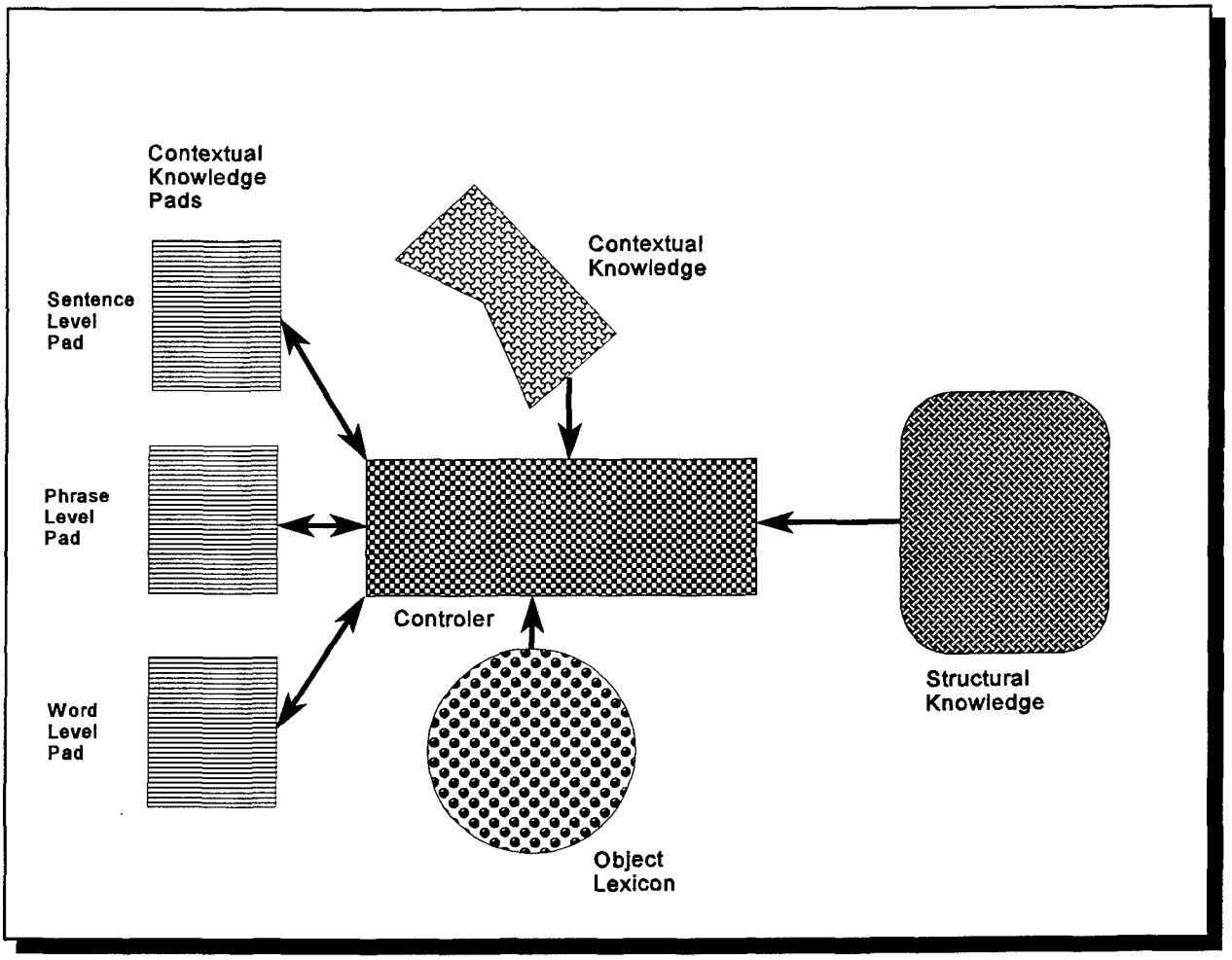


Figure I: System Diagram

Randomly generating a sentence from a template with a fixed vocabulary can yield many unusual and humorous combinations of words. Obviously, not all words can be combined to create a meaningful sentence. The central idea of our system is that the decisions which are made to form one part of the template should limit the available choices for other parts of the template. We view the process of generating a sentence as a series of phrase generation sub-goals. Furthermore, each phrase generation is divided into word generation sub-goals. The generation is performed on a prioritized basis, so that the most important phrases within a sentence, and the most important parts within a phrase are completed first. It is important to point out that most of the constraints used during the generation process are formed dynamically. Each phrase and each component of a phrase can be constructed in a number of ways. However, as parts of each phrase are being constructed, so are the constraints limiting the choices for the subsequent parts. The primary components (figure I) and the objects used to implement our ideas are described below.

System Components

Controller:

As its name implies this component is responsible for controlling the entire generation process. It creates and prioritizes new sentence, phrase, and word objects. Using the contextual knowledge and objects on the contextual knowledge pads the controller builds the constraints used in generation. Finally, it completes the process by forming the words, phrases, and sentences.

Object Lexicon:

This module contains the lexical objects and the knowledge about words that each object contains.

Structural Knowledge:

The structures used in building sentences and phrases are stored in the template and expansion objects in this module.

Contextual Knowledge:

This component is a collection of constraint generating rules.

Contextual Knowledge Pads:

The three areas that comprise the contextual knowledge pads are used to store objects that have already been created. These dynamically created objects along with the contextual knowledge rules regulate the generation process of subsequent parts of the sentence.

System Objects

The system is implemented in an object-oriented environment [1]. The objects utilized in the system are described below:

Template Objects:

The syntactic knowledge obtained from the Benchmark Evaluation Tool is stored in template objects. This knowledge is organized in a tree hierarchy with branch nodes and leaf nodes. The leaf nodes contain the template string, an internal parsed representation of the template, selection flags, and sentence level constraints.

Expansion Objects:

A template is a collection of grammatical patterns, each pattern (such as [NP]) may have multiple ways that can be satisfied [2,10]. The expansion of the patterns found in the templates are stored in expansion objects. These object are also organized in a tree hierarchy. The leaf nodes for these objects are specific parts of speech, for example count nouns or transitive verbs. The branch nodes, from the bottom of the tree up, have increasingly general patterns.

Lexical Objects:

These objects are the words that make up the system's vocabulary. They store syntactic, semantic, and morphological information. The syntactic elements include parts of speech, and any syntactic limitations associated with a word. The morphological information is used to form the word when it deviates from the normal rules of English orthography [3]. The semantic information is used to enforce semantic agreement, it contains concepts or qualities that the word possesses or requires.

Constraint Objects:

Constraint objects are built for every phrase and for every word. They contain the information to be used to limit the generation. These objects have slots for the root, type, number, person, tense, form, case, gender, concept, quality, and quantifier [2,10].

Sentence Objects:

The sentence objects contain the instructions for building the sentence and the final version of the sentence. Sentence object sub-classes include: declarative, imperative, interrogative, and clauses.

Phrase Objects:

The phrase objects are the most complex objects in the system. Most phrase objects inherit from semantic, syntactic, and phrase-type super-classes [6]. For instance, the phrase object, agn-sub-np, is a combination of agent (semantic role), subject (syntactic role), and noun phrase (phrase type). The slots found in any one phrase object will vary, however, any information that could be needed later in the generation process is stored in a phrase object [4].

Word Objects:

The word objects store the expansion pattern for the word, the lexical object selected, and the final version of the word. Each part of speech recognized in the system has a word object sub-class.

3.2.2 Templates and Expansions

The template and expansion objects store the structure upon which a sentence is constructed. They serve as a blueprint and a scaffold for the sentence. The template objects contain the instructions for building a sentence from phrases, while the expansion objects contain the instructions for building a phrase from words. The knowledge in both template and expansion objects is distributed in a tree hierarchy with more specific knowledge contained at the roots. This organization allows for a variety of general requests, while still maintaining the ability for specific requests .

The template objects consist of a textual template, the sentence type, selection flags for the NLP system classes, the template parse, and any sentence level constraints.

The template parse, which is stored in the template leaf-node object, is a generalized representation of the phrases within a template. In this representation, a sentence is viewed as a collection of phrases. Each phrase has semantic knowledge as well as the expansion objects which are used in generation process. The semantic knowledge describes the role each phrase plays within the sentence and is used to avoid structural ambiguity and to prioritize the phrase generation within a sentence. The possible semantic roles are *agent* (subject), *action* (verb), *patient* (direct-object), *recipient* (indirect-object), *attribute* (adjectival), *manner* (adverbial), *auxiliary* (auxiliary verb) and *literal.* For the modifiers (attribute and manner) additional information about what they modify and the type of phrase used for the modification is required.

The system provides a template to match various types of selection requests. If the NLP class is specified, the system will use the selection flags to prune the tree so that only those templates appropriate to the selected class will be available. A request can be made for any node on the tree. If the requested node is a leaf node the template information will be returned. However, if the requested node is a branch node, one of its children will be selected at random. This process will continue recursively until the selected child is a leaf node.

The expansion objects contain the grammatical patterns used to build phrases. The system fills an expansion request by returning a flat list of all possible expansions of the input. Each element of this list is a list of expansion leaf-

node objects. The expansion of a branch-node proceeds by expanding each of its children nodes and then combines them for *and* branches or joining them for *or* branches [9]. The expansion process continues recursively until all branch-nodes are converted into lists of leaf-nodes.

3.2.3 Lexicon

The lexicon contains a collection of lexical objects which store syntactic, semantic, and morphological information. Furthermore, the objects in the lexicon are divided into subcategories based on their part of speech. The object categories include: noun, verb, adjective, adverb, preposition, pronoun, article, and conjunction.

The syntactic information includes the part of speech, the subtype of the word (such as count, mass, or proper for nouns), and any syntactic restrictions. For example, proper nouns are restricted to having a single number. The morphological information contains exceptions to the normal formation rules of the English language [3]. This includes tense formation for verbs, plural formation for nouns, comparative and superlative formation for adverbs and adjectives, and case and gender formation for pronouns. For example, the object for the verb "give" has a past tense form "gave" stored in it.

The lexical objects also store the semantic information that is used to enforce semantic agreement. This information varies for different parts of speech and is as follows:

Nouns & Pronouns:

The nouns and pronouns have two slots for concepts and qualities [8]. The concept slot contains a list of broad categories in which the noun or pronoun can be classified. The quality slot contains a list of attributes on which the noun or pronoun can be modified.

Verbs:

Verbs have the following additional slots: agent, patient, recipient, and quantifiers [8]. The agent, patient, and recipient slots contain semantic concepts that must be present in the candidate nouns. The quantifier slot contains a list of attributes on which the verb can be modified.

Adjectives & Adverbs:

Adjectives and adverbs each have one additional slot that is used to ensure agreement with the object or action they modify. The Adjectives have a qualifier-list slot which contains a list of attributes they can modify. Likewise, the adverbs have a quantifier-list slot which contains a list of attributes they can modify [8].

Prepositions:

Prepositions have one additional slot, concept-list, which contains concepts that must be present in the head noun of the preposition phrase [8]. Agreement between prepositions and the nouns or verbs they modify will be enforced through the preposition's subtype and the noun's quality slot or the verb's quantifier slot.

3.2.4 Constraints & CLOS

The constraints built during the formation of phrases and words are vital to the functionality of this system, for without them the system would be merely an over elaborate method of generating random sentences. This section describes the type of constraints and the method of their implementation.

Constraints, both static and dynamic arise from almost all components of the system. Static constraints come from template objects (in the form of sentence level restrictions), from the expansion objects (in the way they request specific words, word subtypes, a specific tense, or form), and from lexical objects (through syntactic restrictions). Constraints are also added dynamically during the interaction of sentence, phrase, and word objects according to the rules in the contextual knowledge component.

Before discussing the implementation of these dynamic constraints, we need to explain the behavior of CLOS *(COMMON LISP Object System)*, an object-oriented extension of COMMON LISP. CLOS supports the object-oriented concept of polymorphism through a mechanism of methods. Generic functions are functions whose behavior can vary based on the objects they receive in their parameter list [6,11]. They have a single Primary-method which gets executed regardless of parameter values, and optional Before-methods and After-methods which are executed conditionally before and after the primary method when the parameter objects, or any superclasses of the objects, are of the type specified. For instance, the generate-word function in our system receives as parameters a phrase object, a word object, a word constraint object, and the phrase level knowledge pad. This function has several Before-methods which execute depending on the type of objects received in the first two parameters.

As an example, lets look at the situation where the generate-word function is called with a phrase object of agn-sub-np and a word object of noun. This parameter combination triggers a Before-method that is looking for a noun-phrase object and a noun word object. Therefore, before the Primary-method can be executed, the Before-method must be executed. Notice that (Figure II) the phrase object, agn-sub-np, inherited the noun-phrase type from one of the superclasses from which it is composed. Because of this inheritance some parameter combinations may cause multiple Before- or After-methods to be executed.

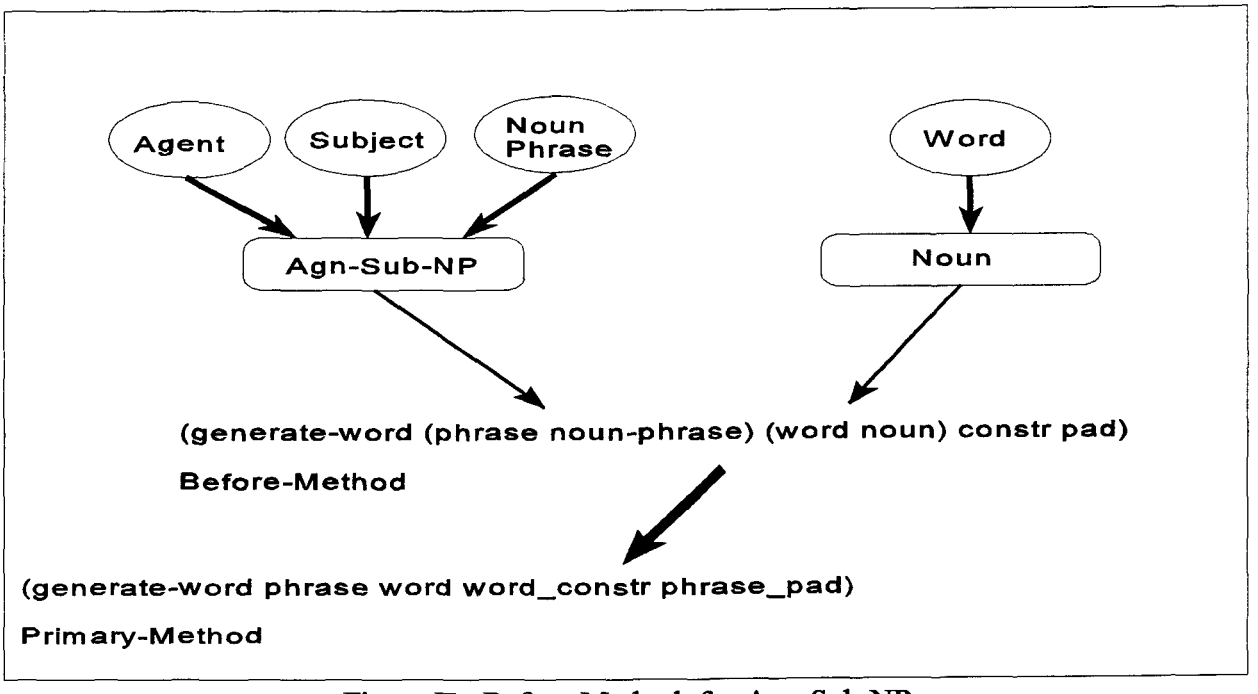


Figure II - Before-Method for Agn-Sub-NP

The two main functions of our system, generate-phrase and generate-word, are both implemented as generic functions. It is through Before-methods and After-methods that the dynamic constraints are built. When these functions are first called the constraint objects in their parameter lists are empty. Before-methods examine other objects in the environment and build the constraints according to the needs of the objects that are being generated. For example, before a agn-sub-np phrase object is created the number, person, and concept slots of the phrase constraint are copied from the act-pred-vp object that had been previously created. Before-methods are also used to copy the static constraints to the constraint objects. After-methods are used to store information for future reference. For example, after the main verb has been selected the number, person, tense, agent, patient, and recipient concepts are stored in the phrase object for future use in constraint building.

3.2.5 System Flow

This section describes the function calls made by the controller in generating a sentence. The next section provides two detailed example to further explain the process.

Sentence Level

After a template has been selected the appropriate sentence type subclass is built using the textual template and the template parse from the template object. This sentence is input to the function generate-sentence which returns the object with the sentence string slot filled in. The textual template and the sentence string are displayed as the system's output.

The generate-sentence function transforms the template parse into a list of phrase objects. These phrase objects are then sorted into a list of goals. Each subgoal phrase is formed by the function generate-phrase. After all the phrases have been generated, the form-sentence function builds the sentence string. The completed sentence object is then returned.

The order of the phrases is based on their semantic roles and is as follows: action, agent, patient, recipient, attribute, manner, auxiliary, literal.

Phrase Level

The build-phrase function takes the phrase information in the parsed template and builds the appropriate phrase object. This information includes the semantic role and an expansion object. For attribute or manner semantic object, it also includes the object being modified and the type of modifying phrase. During the construction of this phrase object, the expansion object is transformed into the list of possible expansions.

The generate-phrase function selects one of possible expansions randomly and transforms it into a list of word objects. These word objects are ultimately sorted into a list of goals, based on the type of phrase. Each word in each goal is formed through the function generate-word. After all the words have been generated, the form-phrase function builds the phrase string. The completed phrase object is then returned.

Word Level

Build-words creates a list of word objects by simply copying each expansion leaf-node object in the phrase's expansion list to a newly created word object.

The generate-word function receives the current phrase object, the current word object, a constraint object, and the phrase-level knowledge pad as input. Before any code is executed in the primary-method the multiple Before-methods are evoked to build the complete constraint object. Included in this process is the selection of those lexical entries which match the part of speech for the target word. Next, all the constraints are applied in order to select only those words that can pass the filter conditions from the selected lexical list. One of the candidates is then chosen at random, the word string is formed and the modified word object is returned.

The following rules illustrate how a regular verb is formed [2].

Present tense: 3rd Pars. Sing : root + "s" otherwise root

Past tense: root + "ed"

Future: "will " + root

Present Progressive: (appropriate present tense of "be") + " " + root + "ing"

Past Progressive: (appropriate past tense of "be") + " " + root + "ing"

Any verb which deviates from the above rules, all irregular verbs expect for the verb "to be", will have an entry in the morphology slot that will explicitly form the verb.

The verb "to be" is formed as follows [2]:

Present tense: 1st Pars. Sing. = "am" 3rd Pars. Sing. = "is" otherwise "are"

Past tense: 1st or 3rd Pars. Sing. = "was" otherwise "were"

Future tense: "will be"

3.2.6 Example 1

To see how a sentence is generated we will walk through the generation processes for the following sentence: "The supervisor hired a new employee". The template for this sentence is: [NP] [Verb] [Det] [Adj] [Noun].

Create sentence object from knowledge stored in template object. Generate a sentence from this object.

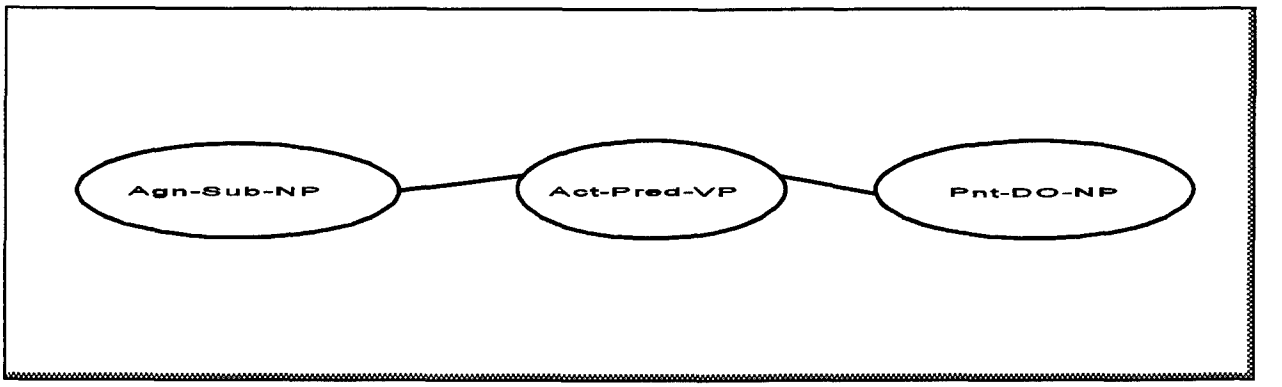
Generate Sentence:

Build phrase objects from the parse stored in the sentence object. For our sample sentence three objects are created: agn-sub-np (agent-subject-noun_phrase), act-pred-vp (action-predicate-verb_phrase), pnt-do-np (patient-direct_object-noun_phrase). Each object includes all expansion possibilities for that object.

Order phrases by priority based on their semantic roles. The ordered phrase list is act-pred-vp, agn-sub-np, pnt-do-np.

Generate each phrase.

Form the sentence by combining the phrases.



Generate Act-Pred-VP:

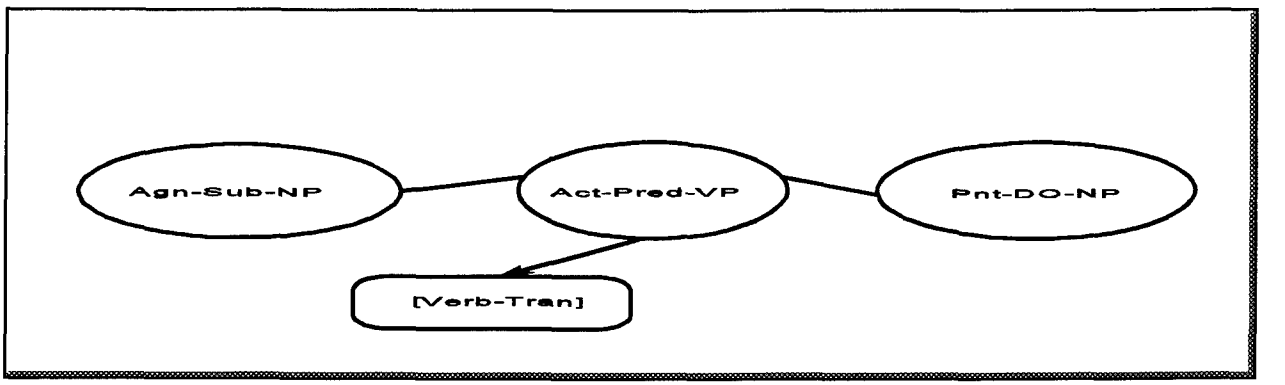
Before: No action, however, sentence-level constraints would be copied to the phrase constraint here.

Primary: Select one of the possible expansions at random. For this template [verb-tran] is the only choice.

Build word object(s) for each expansion element.

Order words by priority based on part of speech. Generate each word.

After: No action taken.



Generate Verb-Tran:

Before: Build word constraint object:

Word List = Verbs

Type = Transitive

Primary: Apply constraints to reduce list of verbs.

Choose word randomly. "hire" is chosen.

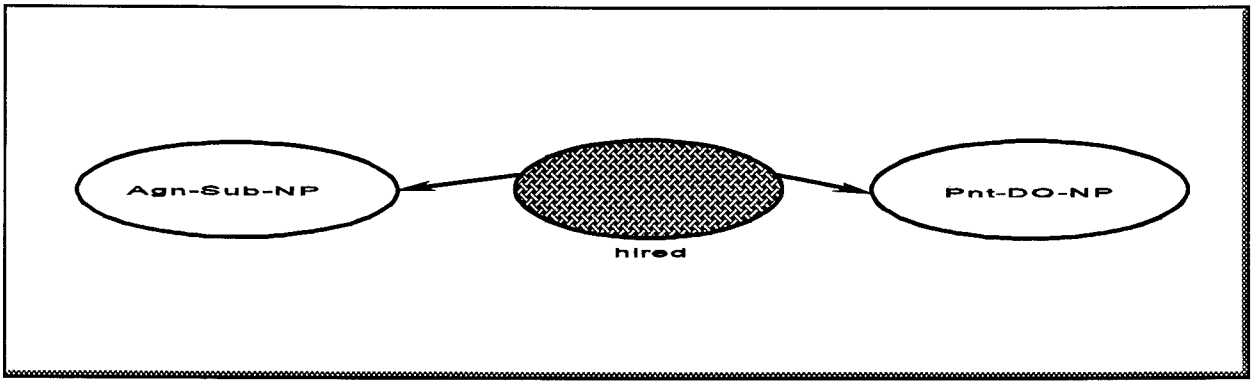
Form word and store in word object

Choose number, person and tense at random. Singular, third, past are chosen.

Use standard formation rules or lexical object morphology to build correct form of the word. The string "hired" is built.

After: Copy predicate, number, person, tense, agent-concepts, patient-concepts and quantifier-list to act-pred-vp object.

The phrase "hired" is formed.



Generate Agn-Sub-NP:

Before: Build Phrase Constraint:

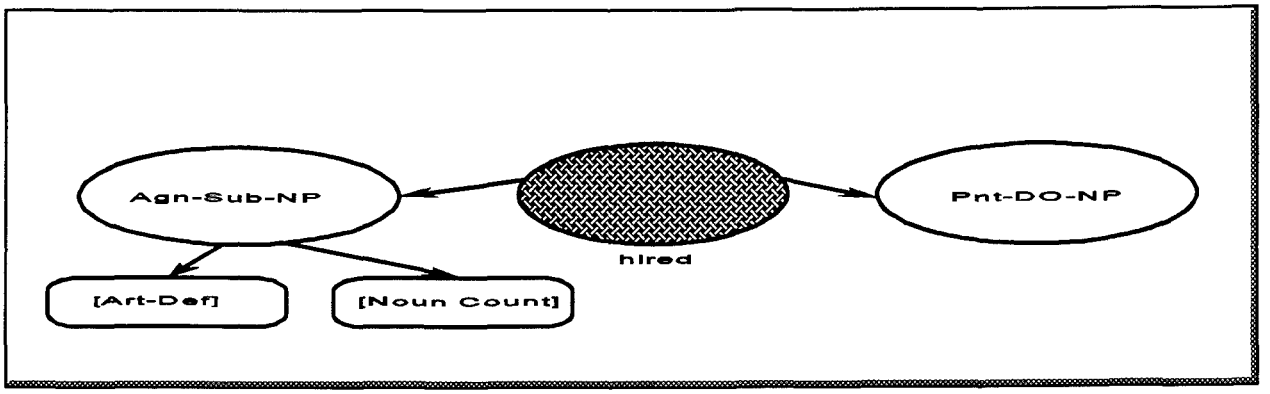
Copy number, person, and agent-concepts from the act-pred-vp object.

Primary: Select one of the possible expansions at random. [art-def] [noun-count] is chosen.

Build word object(s) for each expansion element.

Order words by priority based on part of speech. Generate each word.

After: No action taken.



Generate Noun-Count:

Before: Build word constraint object:

Word List = Nouns

Type = Count

Number = Singular

Person = Third

Concept-List = (Individual Management Personal)

Primary: Apply constraints to reduce list of nouns.

Choose word randomly. "supervisor" is chosen.

Form word and store in word object

Use standard formation rules or lexical object morphology to build correct form of the word. The string "supervisor" is built.

After: Copy subject, number, person, agent-concepts and quality-list to agn-sub-np object.

Generate Art-Def:

Before: Build word constraint object:

Word List = Articles

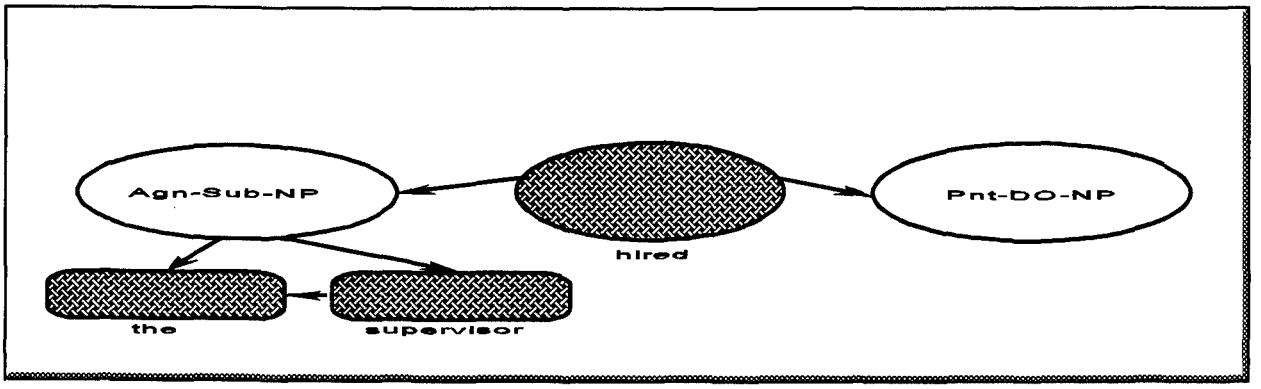
Type = Definite

Primary: Apply constraints to reduce list of articles.

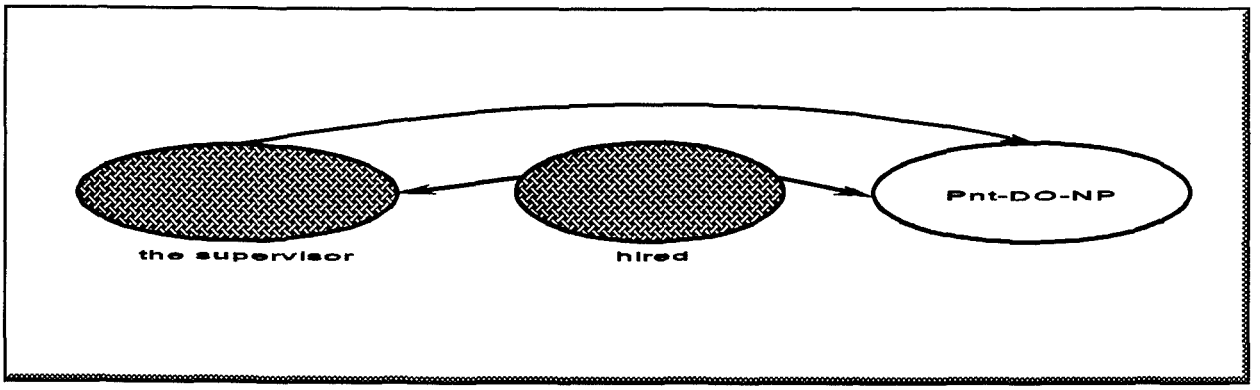
Only the word "the" meets the constraints.

Store "the" in word object

After: No action taken.



The phrase "the supervisor" is formed.



Generate Pnt-DO-NP:

Before: Build Phrase Constraint:

Copy patient-concepts from the act-pred-vp object.

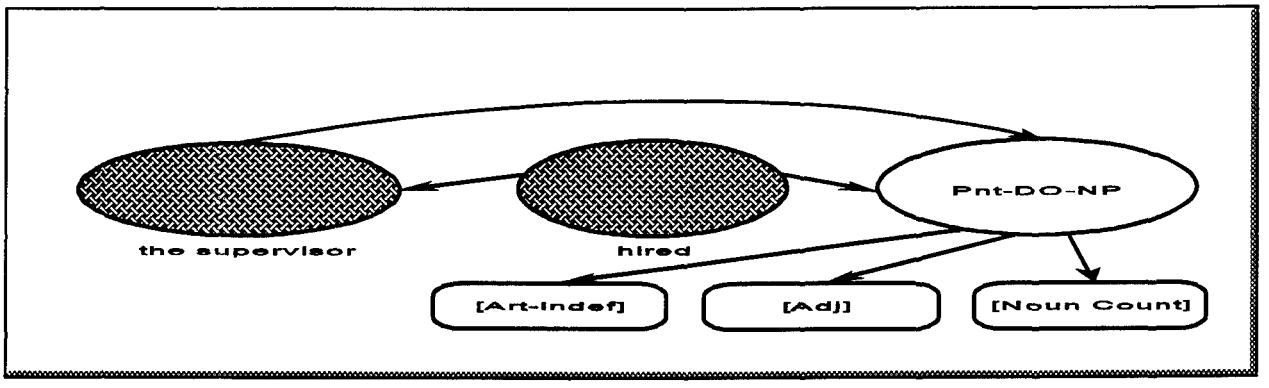
Copy subject (so subject and object aren't the same) from the agn-sub-np object.

Primary: Select one of the possible expansions at random. [art-indef] [adj] [noun-count] is chosen.

Build word object(s) for each expansion element.

Order words by priority based on part of speech. Generate each word.

After: No action taken.



Generate Noun-Count:

Before: Build word constraint object:

Word List = Nouns

Type = Count

Concept-List = (Individual Labor Personal)

Root <> "supervisor"

Primary: Apply constraints to reduce list of nouns.

Choose word randomly. "employee" is chosen.

Form word and store in word object

Choose number, person at random. Singular, third are chosen.

Use standard formation rules or lexical object morphology to build correct form of the word. The string "employee" is built.

After: Copy direct object, number, person, patient-concepts and quality-list to agn-sub-np object.

Generate Adj:

Before: Build word constraint object:

Word List = Adjectives

Type = Qual

Quality-List = (Age Intellect Size Marital-Status Sex Work-Hist)

Primary: Apply constraints to reduce list of adjectives.

Choose word randomly. "new" is chosen.

Store "new" in word object.

After: No Action taken.

Generate Art-Indef:

Before: Build word constraint object:

Word List = Articles

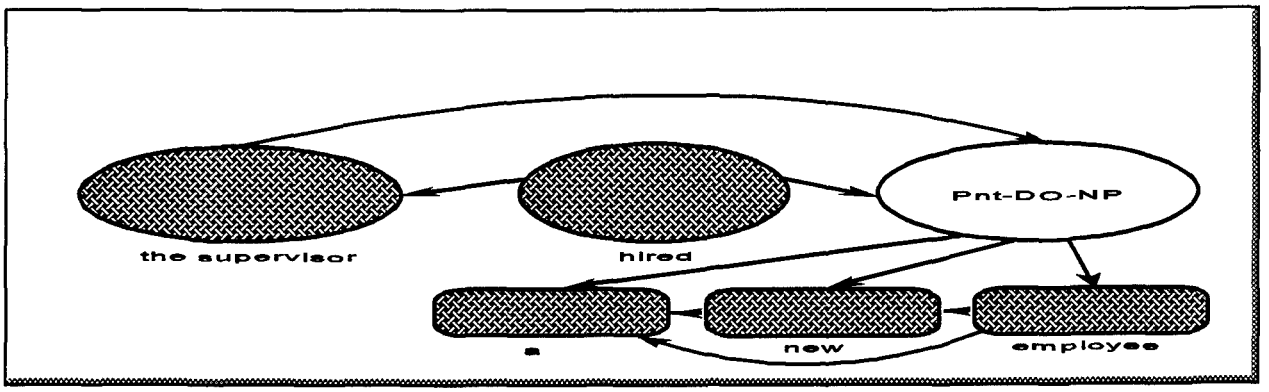
Type = Indefinite

Primary: Apply constraints to reduce list of articles.

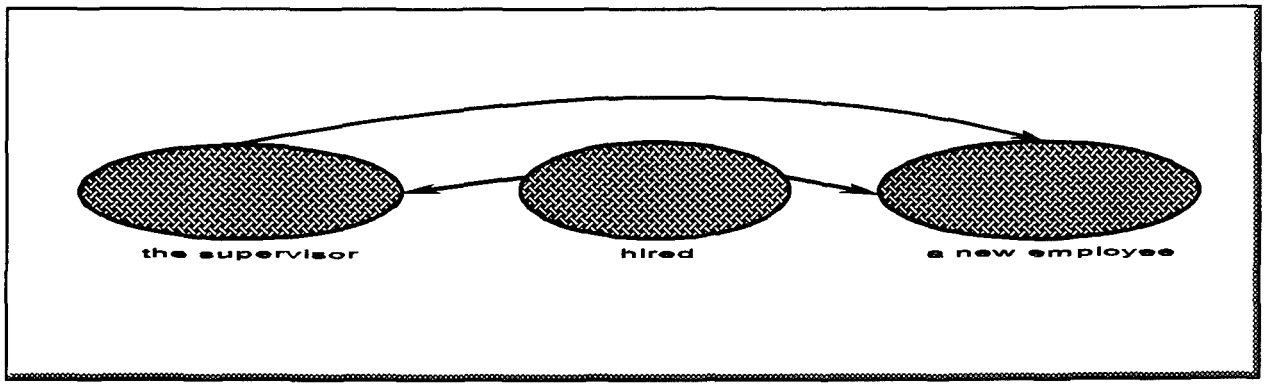
Only the word "a" meets the constraints.

Store "a" in word object

After: No action taken.



The phrase "a new employee" is formed.



Finally, the complete sentence "The supervisor hired a new employee." is generated.

3.2.7 Example 2

Let's now look at the generation of the following sentence: "Who did the supervisor hire?". The template for this sentence is Who [DO-Verb] [NP] [VP].

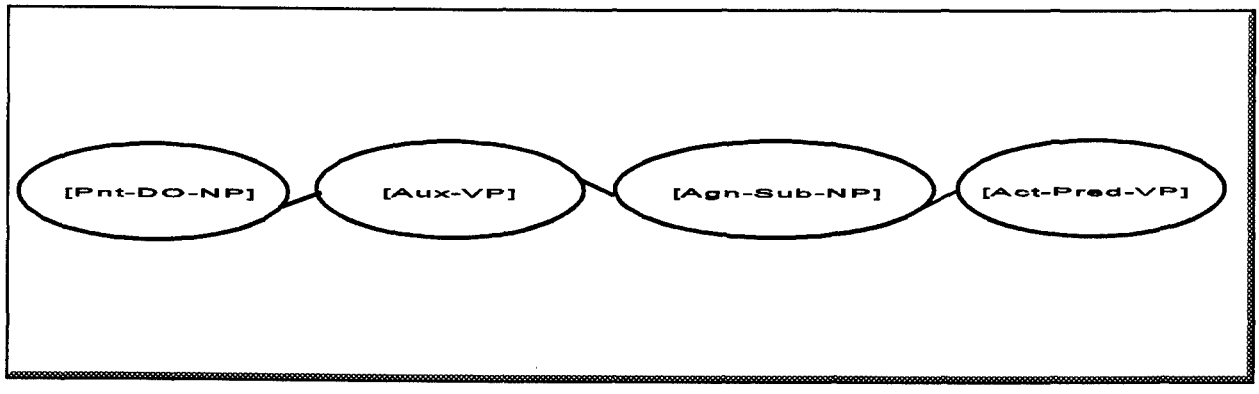
Create sentence object from information stored in template object. Generate a sentence from this object.

Generate Sentence:

Build phrase objects from the parse stored in the sentence object. For this sentence four objects are created: pnt-do-np, aux-vp, agn-sub-np, and act-pred-vp.

Order phrases by priority based on semantic role. The ordered phrase list is act-pred-vp, agn-sub-np, pnt-do-np, aux-vp. Generate each phrase.

Form the sentence by combining the phrases.



Generate Act-Pred-VP:

Before: No action taken.

Primary: Select one of the possible expansions at random. For this template [verb-root] is the only choice.

Build word object(s) for each expansion element.

Order words by priority based on part of speech. Generate each word.

After: No action taken.

Generate Verb-Root:

Before: Build word constraint object:

Word List = Verbs

Type = Transitive

Tense = Root

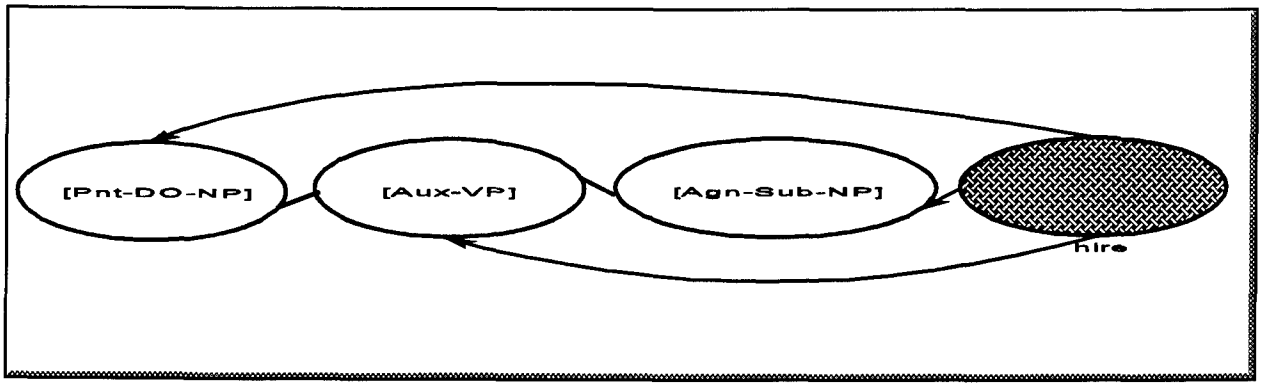
Primary: Apply constraints to reduce list of verbs.

Choose word randomly. "hire" is chosen.

Store root form "hire" in word object.

After: Copy predicate, agent-concepts, patient-concepts and quantifier-list to act-pred-vp object.

The phrase "hire" is formed.



Generate Agn-Sub-NP:

Before: Build Phrase Constraint:

Copy agent-concepts from the act-pred-vp object.

Primary: Select one of the possible expansions at random. [art-def] [noun-count] is chosen.

Build word object(s) for each expansion element.

Order words by priority based on part of speech. Generate each word.

After: No action taken.

Generate Noun-Count:

Before: Build word constraint object:

Word List = Nouns

Type = Count

Concept-List = (Individual Management Personal)

Primary: Apply constraints to reduce list of nouns.

Choose word randomly. "supervisor" is chosen.

Form word and store in word object

Choose the number and person at random. Singular, third is chosen.

Use standard formation rules or lexical object morphology to build correct form of the word. The string "supervisor" is built.

After: Copy subject, number, person, agent-concepts and quality-list to agn-sub-np object.

Generate Art-Def:

Before: Build word constraint object:

Word List = Articles

Type = Definite

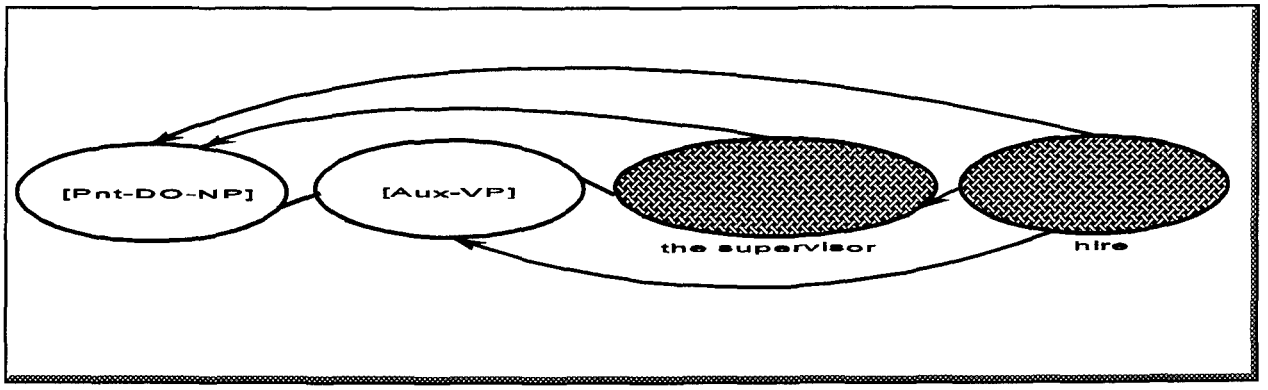
Primary: Apply constraints to reduce list of articles.

Only the word "the" meets the constraints.

Store "the" in word object

After: No action taken.

The phrase "the supervisor" is formed.



Generate Pnt-DO-NP:

Before: Build Phrase Constraint:

Copy patient-concepts from the act-pred-vp object.

Copy subject from the agn-sub-np object.

Primary: Select one of the possible expansions at random. For this template [who-pro] is the only choice.

Build word object(s) for each expansion element.

Order words by priority based on part of speech. Generate each word.

Form the phrase by combining the generated words. Phrase is "who".

After: No action taken.

Generate Who-Pro:

Before: Build word constraint object:

Word List = Pronouns

Root = "who"

Concept-List = (Individual Labor Personal)

Root <> "supervisor"

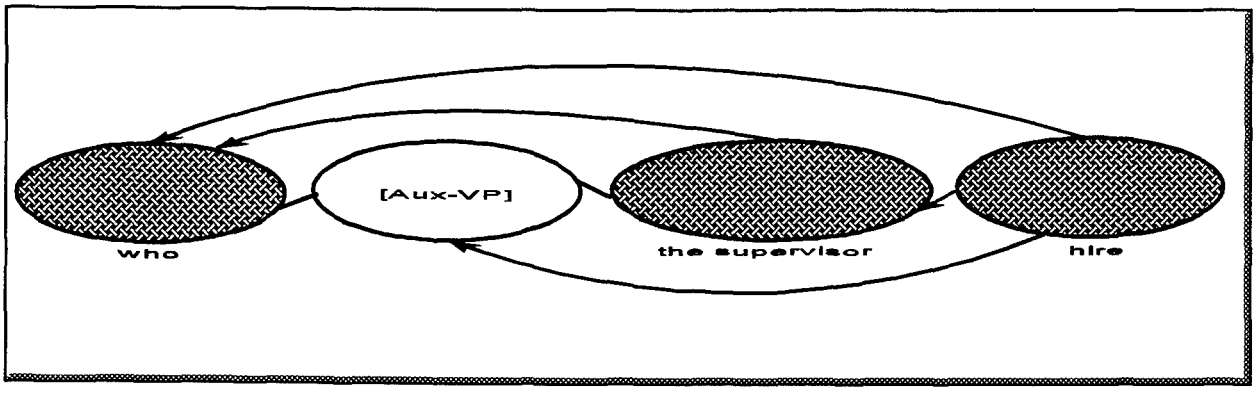
Primary: Apply constraints to reduce list of pronouns.

Only the word "who" meets the constraints.

Store "who" in word object

After: No action taken.

The phrase "who" is formed.



Generate Aux-VP:

Before: No action taken.

Primary: Select one of the possible expansions at random. For this template [DO-Verb] is the only choice.

Build word object(s) for each expansion element.

Order words by priority based on part of speech. Generate each word.

Form the phrase by combining the generated words. Phrase is "did".

After: No action taken.

Generate DO-Verb:

Before: Build word constraint object:

Word List = Verbs

Root = "do"

Primary: Apply constraints to reduce list of verbs.

Only the word "do" meets the constraints.

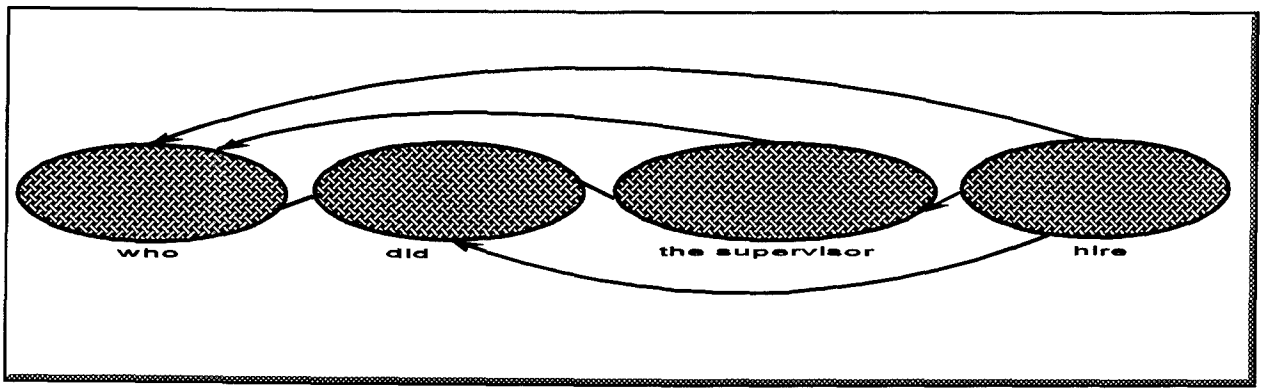
Form word and store in word object.

Choose tense at random. Past is chosen.

Use standard formation rules or lexical object morphology to build correct form of the word. The string "did" is built.

After: No action taken.

The phrase "did" is formed.



Finally, the complete sentence "Who did the supervisor hire?" is generated.

4.0 Results

The system can operate on one of five modes. Each mode represents a class of NLP system and is an input to the system. This input is utilized to limit the template choices at the top level and does not alter the functionality of the system in any way. The sentences are generated to satisfy the semantic, pragmatic, and contextual constraints. The semantic and pragmatic constraints are stored as static objects in contextual knowledge base as well as in object lexicon. The contextual constraints are also stored as objects in contextual knowledge pad but are generated dynamically during the generation of each sentences. These constraints are only kept temporarily during each sentence generation and are erased from the contextual knowledge pad once the sentence is completed.

This generation system is a tool to assist the human evaluator with the evaluation process. The human evaluator indicates the class of NLP system being evaluated and the system generates semantically meaningful applicable test sentences. The human evaluator can then score the NLP system's response. Our objective has been to increase the human evaluator's productivity by automating the test sentence generation phase of evaluation process. This way, the evaluator can spend more time analyzing and scoring the NLP system's response rather than forming sentences that are applicable to the system.

Due to our highly modular design which is based on an object-oriented framework, adding additional functionalities to the existing system is extremely simple. With relatively a few number of words, approximately 100, we were able to demonstrate the power of our system. Expanding the object lexicon, by adding new words, will dramatically enhance the strength and flexibility of the system. The system operates in real time. The average generation time for each test sentence, after all superclass objects have been instantiated, is approximately 1.3 seconds. The following is a sample set of test sentences along with their syntactic and semantic templates the system generated.

[PP] [NP] [VP]

(Mnr-Mod-PP) (Agn-Sub-NP) (Act-Pred-VP)

On Monday Mary Smith produced the computers.

[WH-Word] ([NP]) [Verb] [NP] [PP]

(Agn-Sub-NP) (Act-Pred-VP) (Pnt-DO-NP) (Mnr-Mod-PP)

Who sells the books to Texas?

[NP] [Verb] ([Det]) [Count Noun]

(Agn-Sub-NP) (Act-Pred-VP) (Pnt-DO-NP)

The directors hired a salesperson.

[Verb] ([Det]) [Count Noun]

(Act-Pred-VP) (Pnt-DO-NP)

Hire the salesperson.

[NP] [Verb] [NP] [PP]

(Agn-Sub-NP) (Act-Pred-VP) (Pnt-DO-NP) (Mnr-Mod-PP)

The salesperson sells the book on Monday.

[NP] [BE-Verb] [Adj]

(Agn-Sub-Np) (Act-Pred-VP) (Atr-Mod-Adjp)

Jane Doe is large.

Who [Verb] [NP]

(Agn-Sub-NP) (Act-Pred-VP) (Pnt-DO-NP)

Who will hire the engineers?

List [NP] [Rel Pronoun] [Verb] [NP]

(Act-Pred-VP) (Pnt-DO-NP) (Atr-Mod-RC)

List the presidents who hire the engineers.

[NP] [Verb] anyone [PP]

(Agn-Sub-NP) (Act-Pred-VP) (Pnt-DO-NP) (Atr-Mod-PP)

John Doe will fire anyone in Austin.

List [NP] [Rel Pronoun] [Verb +pres-prog] [NP]

(Act-Pred-VP) (Pnt-DO-NP) (Atr-Mod-RC)

List the directors who are hiring the engineers.

[DO-Verb] [NP] [VP]

(Aux-VP) (Agn-Sub-NP) (Act-Pred-VP) (Pnt-DO-NP)

Did the directors fire Mary Smith?

[Det] [Adj] [Noun] [VP]

(Agn-Sub-NP) (Act-Pred-VP) (Pnt-DO-NP)

The smart manager hires an engineer.

5.0 Conclusions

Evaluating an NLP system is a complicated task. There are several areas in which NLP systems can be evaluated. They include: a) linguistic competence, b) end user issues such as reliability and likability, c) system development issues such as maintainability and portability, and d) intelligent behavior issues such as learning and cooperative dialogue. The Benchmark Evaluation Tool was very thorough in its coverage of the English language, but we feel that its expectations for linguistic competence and for domain independence are not attainable given the current state of technology. The task in the first phase of our project was to compile a collection of sentences from the Benchmark Evaluation Tool that were appropriate to each of five NLP systems classes. Based on our previous experience with the Benchmark Evaluation Tool limiting the scope of the evaluation appeared to be the most realistic and constructive approach.

Therefore, in the first phase of this project, we attempted to define boundaries for each of the five classes of NLP systems. In the second phase of our project we attempted to automate the sentence generation process. We designed and implemented an object-oriented framework for generating non-causal sentences. In this system, syntactic and semantic knowledge were embedded concurrently in our design. Each word-object, for instance, has local static knowledge which enables it to make decision. Hence, each word is an independent knowledge source. In addition, contextual constraints are generated dynamically which will limit the available choices for the subsequent parts of the sentence. This approach, which mimics the way human form sentences, worked well and showed much future promise. This is the only way to preserve the cohesiveness and to maintain the compatibility of parts of each sentence. The object-oriented viewpoint proved extremely beneficial in coding rules, in that the same object could be viewed in a variety of ways. The modular design allows for the addition of lexical entries and templates that can increase the expressive power of the system with little or no changes to the core code.

Our objective here has been to automate the sentence generation process. Hence, we have concentrated on generating non-causal sentences. We believe that we have broken new ground. A natural extension of this work is to modify the system to generate causal sentences, sentences which express a desired meaning.

The proposed system would add another layer on top of the current system. This layer would receive as input a collection of objects, each object expressing a segment of the desired intention, and collectively expressing the complete theme or intention. This information is used to select the appropriate sentence structure and to construct the proper thematic constraints and would then serve as input to the current system to guide the generation process. For instance, the following represents a partial overview of three objects: *(agent john) (action ptrans past) (recipient store (category old))*. To express the desired intention, the following surface structure: [NP] [verb] [PP] along with the following semantic structure: (Agn-Sub-NP) (Act-Pred-VP) (Pnt-DO-NP) may be selected. These structures along with the three objects above guide the system to generate the sentence: 'John went to the old store.' This extension would require very little change in the current system. The mechanism for allowing sentence level constraints already exists in the current system. The thematic constraints would simply need to be expressed as sentence level constraints in order to be passed throughout the system.

The system developed here is the first step in completely automating the entire evaluation process. The entire evaluation process of an NLP system performed by a human evaluator requires three stages. In stage one, a test sentence is generated; in stage two, the NLP system's response is analyzed; and in stage three, the NLP system's response is scored. The current state of our generation system performs stage one of the evaluation process. We plan to continue this effort by extending the system to generate causal sentences and to perform stage two and three of the evaluation process.

6.0 References

[1] Booch G. 1991. *Object Oriented Design with Applications.* Benjamin/Cummings Publishing Company.

[2] Fowler H., 1980. *The Little, Brown Handbook.* Little, Brown and Company.

[3] Gazdar G., and Mellish C., 1989. *Natural Language Processing in LISP.* Addison-Wesley Publishing Company.

[4] Hirst G., 1987. Semantic *Interpretation and the Resolution of Ambiguity.* Cambridge University Press.

[5] Kaikhah, K., 1992. *An Investigation of the Benchmark Evaluation Tool.* Technical Report, Air Force Office of Scientific Research (AFOSR), Summer 1992.

[6] Keene S., 1989. *Object-Oriented Programming in Common LISP.* Addison-Wesley.

[7] Neal J., Feit E., Funke D., and Montgomery C. 1992. Benchmark Investigation/Identification Program. Technical Report, Calspan Advanced Technology Center.

[8] Sowa J., 1988. Using a Lexicon of Canonical Graphs in a Semantic Interpreter in *Relational Models of the Lexicon: Representing Knowledge in Semantic Networks.* ed Evans, M.. Cambridge University Press.

[9] Steele G., 1990. *Common LISP - Second Edition.* Digital Press.

[10] Thomson A., and Martinet A., 1986. *A Practical English Grammar - Fourth Edition.* Oxford University Press.

[11] Winston P., and Horn B., 1989. *LISP - Third Edition.* Addison-Wesley Publishing Company.

SYSTEMS ANALYSIS FOR A PHOTONIC DELAY LINE

Evelyn H. Monsay
Associate Professor
Department of Physics

Le Moyne College
1419 Salt Springs Road
Syracuse, New York 13214

Final Report for:
AFOSR Research Initiation Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, D.C.

March 1994

SYSTEMS ANALYSIS FOR A PHOTONIC DELAY LINE

Evelyn H. Monsay
Associate Professor
Department of Physics
Le Moyne College

Abstract

A photonic means of generating the element-to-element phase shifts required for the operation of a true time delay radar beamformer was developed previously by Toughlian and Zmuda for systems operating in the VHF frequency range, and extended by Monsay for operation in the radar C band. The photonic beamformer is based on an acousto-optic (AO), heterodyne optical system, in which the reference beam acts as a probe of the acoustic wave in the AO cell. The time delay of the acoustic waveform "read out" by the optical probe from various locations along the length of the AO cell can then be imposed directly on the microwave elements of the radar beamformer, providing the correct phase for each element at whatever frequency drives the AO cell.

In this work, fundamental systems-oriented calculations are reported. Results for both optical specifications and radar performance specifications were obtained, based upon a mathematical model of the photonic delay line previously developed by the author. These results address the use of the photonic delay line in a standard radar beamformer. Future work will use these results in evaluating the potential performance of the photonic delay line in adaptive beamforming and other applications.

SYSTEMS ANALYSIS FOR A PHOTONIC DELAY LINE

Evelyn H. Monsay

I. INTRODUCTION

Superior performance of wideband phased-array radar antennas requires true time delay phase shifts between elements. However, building a radar system with conventional true time delay phase shifters is costly and the result is too heavy for many airborne or satellite applications. Previous photonic methods of generating true time delay lines have relied on varying lengths of optical fiber for each element. However, such methods are cumbersome and difficult to implement due in part to large optical losses.

Within the past few years, Toughlian and Zmuda [1-4] have developed an RF true time delay phase shifter based on an acousto-optic (AO) heterodyne optical system. Sumberg [5] has investigated means for integrating the optical system. In Monsay [6] and Monsay, Baldwin, and Caccuitto [7], the results of using a dichromatic laser, demonstrated the potential of the photonic beamformer for providing true time delays for high frequency (C band) radar systems.

Various potential operational problems were also considered [6]. Since the photonic delay line is based on a heterodyne optical system, good laser noise suppression is automatically built in. In low-frequency (VHF) systems, both reference and signal beams carry the same noise disturbances which then cancel out upon coherent interference. The reference and signal beams in the dichromatic laser or coherent laser system required for high-frequency (C-band, etc.) operation will, to the degree of coherence exhibited by the particular laser or laser system, also tend to suppress laser jitter and drift.

However, it was suggested [6] that the most important potential problem for the photonic delay line would arise from pointing errors of the laser beam along the acoustic wave in the AO cell. Such pointing errors could arise from mode hopping in the laser, or from mechanical dithering of the rotating mirror which directs the probe beam. In addition, small, but not negligible, oscillations in the data of phase versus frequency were observed in both low- and high-frequency photonic delay lines, potentially exaggerating the effect of small deviations in pointing angle on the accuracy of the final radar system. These "wiggles" in the data proved to be highly reproducible; they were apparently not random noise, but some true physical effect requiring further study of the optical system.

The study of the diffraction of the optical probe beam, presented in

Reference [8], provided a theoretical understanding of the "wiggles" in the data. In Section II, this mathematical model is reviewed and summarized. In Section III, the optical SNR, a critical element in the analysis of the photonic delay line for application to radar systems, is derived and evaluated. In Section IV, the mathematical model of the photonic delay line is used to predict several system-level performance parameters - beampattern, beamwidth, sidelobe level, and array gain - for radar systems based on the photonic approach. Conclusions and recommendations are presented in Section V.

II. PHOTONIC DELAY LINE

Figure 1 presents the architecture of the basic photonic true time delay line, using a dichromatic laser as for a high-frequency radar system. The optical configuration is a standard heterodyne system with signal and reference beam frequencies of f_S and f_R, and amplitudes A_S and A_R, respectively. A lens of focal length F transforms the point-like probe illumination in the AO cell into a plane wave. The AO cell is used in the Bragg mode of operation. With AO cell RF frequency offset f_M, the resulting optical beams are

$$E_{SIGNAL}=A_S\exp\left[j\left(2\pi\left(f_S+f_M\right)t+2\pi f_x x-\frac{\pi x_p^{\,2}}{\lambda F}+\phi\right)\right] \tag{1}$$

$$E_{REFERENCE}=A_R\exp\left[j2\pi f_R t\right] \tag{2}$$

where ϕ is an optical phase shift representing the accumulated phase difference of the two beams as they propagate through the system. In Equation (1), f_x is the spatial frequency of the heterodyned output, where there is a unique identification of spatial frequency with mirror position angle θ, and so with the location of the probe beam in the AO cell, x. The location x corresponds to a unique time delay in the acoustic time waveform. (The variable x_p runs over the face of the photodetector.)

The heterodyned output, ie, the coherent addition of the two optical beams, has the units of optical power and is given by the square of the sum of the two complex beam amplitudes. This operation results in a constant dc background illumination plus the system output signal with frequency dependence given by

$$I_S(t) \sim A_R A_S \cos\left[2\pi\left(f_x x + [\Delta f + f_M]\, t\right) - \frac{\pi x_p^2}{\lambda F} + \phi\right] \quad (3)$$

In Equation (3), the self-heterodyne frequency of the laser output, $\Delta f = (f_S - f_R)$, appears, modulated by the RF frequency f_M. Hence, the system operates in a band defined by lower frequency Δf and upper frequency $(\Delta f + f_{M(max)})$. Note that any additional phase shift imposed on either beam, here given by

$$\Phi = 2\pi\left(f_x x + [\Delta f + f_M]\, t\right) + \phi \quad (4)$$

comes through directly in the final detected optical signal.

Figure 2 indicates the selective action of the system optics within the AO cell, where a specific phase shift can be "picked off" from the acoustic wave in the cell. The rotating mirror will establish a location, and so a particular time delay, along the acoustic wave relative to its launch point in the cell. As, f_M changes, the phase of the acoustic wave at that fixed point in the AO cell will also change, such that a linear dependence of optical phase on RF frequency - the hallmark of the true time delay phase shift - is established. Hence, for a wideband RF signal with upper and lower frequencies f_U and f_L, respectively, a true time delay phase shift for each frequency component of the RF signal will be available with varying magnitude dependent on which particular point within the AO cell is chosen by one, or more (for multiple antenna elements), rotating mirror(s).

The previous discussion is based on the assumption that the optical probe of the AO cell is performed by a point-like beam. The resulting dependence of read-out phase Φ on signal frequency $f_{sig} = (\Delta f + f_M)$ is then perfectly linear [3]:

$$\Phi = \frac{2\pi x f_{sig}}{v} \quad (5)$$

However, the data on $\Phi(f)$ *for fixed x* from the demonstration of a high-frequency (Figure 3) [6,7] photonic delay line shows very regular oscillations superimposed on the correct linear dependence. Additional study of the photonic delay line architecture with a wide range of focal lengths, laser frequencies, and other parameters indicated that the oscillations were relatively independent of the parameter values over a sizeable range [9].

Two candidate effects could possibly create these oscillations: the windowing effect of the finite aperture photodetector, and the diffraction

from a finite-sized (non-infinitesimal) probe beam. Both effects were studied and found to contribute to nonlinear behavior in Φ.

The original analysis of the photonic delay line assumed an infinitesimal, point-like probe beam reading out the phase of the acoustic wave in the AO cell. However, realistically, the ability to focus the probe beam is limited by diffraction effects. As calculated in Reference [8], the effect of a realistic, finite aperture probe beam is given by the factor

$$\Phi(x,\omega)=atan\left[\frac{\sin(\beta x)\int_{-\frac{1}{d_0}+\beta}^{\frac{1}{d_0}+\beta}\frac{\sin(au)}{u}\cos(xu)\,du-\cos(\beta x)\int_{-\frac{1}{d_0}+\beta}^{\frac{1}{d_0}+\beta}\frac{\sin(au)}{u}\sin(xu)\,du}{\sin(\beta x)\int_{-\frac{1}{d_0}+\beta}^{\frac{1}{d_0}+\beta}\frac{\sin(au)}{u}\sin(xu)\,du+\cos(\beta x)\int_{-\frac{1}{d_0}+\beta}^{\frac{1}{d_0}+\beta}\frac{\sin(au)}{u}\cos(xu)\,du}\right] \quad (6)$$

Figure 4 shows a comparison of $\Phi(x,\omega)$ versus ω for two largely separated values of x, and should be compared with the data in Figure 3.

Although, there will be diffraction effects in the output signal of the photonic delay line due to the finite aperture of the photodetector, it was shown [8] that these effects are constant with respect to frequency, once x is fixed. Hence, these effects would be scaled out of the data on $\Phi(x,\omega)$ versus ω once the mirror position is fixed, and so would not be useful in explaining the data of Figure 3. However, the finite aperture of the photodetector does have an effect on the signal-to-noise ratio (SNR) of the optical system, as will be shown in the next Section.

III. SIGNAL-TO-NOISE RATIO (SNR) OF THE OPTICAL SYSTEM

The general definition of the signal-to-noise ratio (SNR) of a system [10] is given by

$$SNR=\frac{Signal.Power}{\sum_{i=1}^{n}Noise.Power_i} \quad (7)$$

for a system with n different noise sources. For optical power P incident homogeneously over the surface of a photodetector of area A and responsivity S, the photocurrent generated is i=SP. In terms of optical intensity I=P/A, the photocurrent becomes i=SIA. The signal power is given by $\langle i^2 \rangle R_L$, where $\langle\rangle$ denotes the statistical mean, and R_L is the detector load resistance.

The three primary noise sources for a heterodyne optical system include the dark current of the photodetector, the quantum (or "shot") noise, and the thermal (or "Johnson") noise. Letting B denote the post-detection bandwidth, the SNR becomes

$$SNR = \frac{\langle i^2 \rangle R_L}{2eB(i_d + \langle i \rangle)R_L + 4kTB} \tag{8}$$

where e is the electron charge, k is Boltzmann's constant, and T is temperature in degrees Kelvin.

For a heterodyne optical system, the average mean-squared signal current will depend on the product of the average reference beam optical power with the average signal beam power, as is evident from Equation (3). These optical powers will, in turn, depend on the transmission losses, T_i, through the optical system due to lenses, reflection loss, transmission through the AO cell, etc. In the simple system depicted by Figure 1, the reference beam will experience power loss due to transmission through two polarized beam splitters and reflection from two mirrors. The signal beam experiences loss from two polarized beamsplitters, the rotatable mirror, two thin lenses, and the AO cell. Hence, the optical powers received at the photodetector are at most given by

$$P_S = P_{laser} T_S T_{PBS}^2 T_M^2 T_{AO} \tag{9}$$

$$P_R = P_{laser} (1 - T_S) T_{PBS}^2 T_M^2 \tag{10}$$

where T_S is the fraction of the laser power diverted into the signal beam. Typical values of the parameters needed to evaluate the optical SNR are presented in Table 1. Using those values, the resulting SNR equals

$$SNR(dB) = 10\log(8.8x10^7) = 79.45dB. \tag{11}$$

While Equation (11) indicates a good SNR for the typical photonic beamformer optical system, the finite aperture of the photodetector, found <u>not</u> to affect the value of the phase shift Φ [8], <u>will</u> have an effect on the optical SNR. This effect arises because the light illuminating the

photodetector in a heterodyne optical system is not homogeneous, but instead forms an interference pattern of bright and dark regions. The photodetector, according to its size and location, will intercept a variable amount of optical power. In particular, the interference pattern produced by the photonic beamformer will depend on the chosen RF frequency and on the particular point in the Bragg cell illuminated by the probe beam (and, hence, on the mirror tilt angle [3]).

In order to determine the effect of the finite aperture of the photodetector on optical SNR, Equations (1) and (2) can be summed and squared, yielding the optical power distribution produced by the photonic beamformer:

$$p_{TOT}=p_S+p_R+2\sqrt{p_S p_R}\cos\{2\pi(f_x x+f_{sig}t)-\frac{\pi x_p^2}{\lambda F}+\phi\} \qquad (12)$$

where $f_{sig}=[\Delta f+f_M]$, and the optical power densities in the signal and reference beams - p_S and p_R, respectively - appear. Note that the heterodyne signal term, previously given in terms of optical intensity in Equation (3), appears as the third term in Equation (12). The total optical power collected by the photodetector will be given by the integral of the intensity pattern over the area of the detector. Only the time-varying part of Equation (12) is of importance in calculating the SNR, hence the optical signal power in the heterodyne signal can be written as

$$P_{SIG}=\iint_{DetectorArea} 2\sqrt{p_S p_R}\cos\{2\pi(f_x x+f_{sig}t)-\frac{\pi x_p^2}{\lambda F}+\phi\}dA. \qquad (13)$$

It is natural for this system, in which a one-dimensional "fan" of optical beams leaves the AO cell, to find that only the x-dimension (x_p) need appear in the integrand in Equation (13). The y-integration results in mere multiplication by the width of the photodetector.

The cosine in the integrand in Equation (13) can be expanded, and the time dependence of the argument separated out. Then, using the identity

$$A\cos(\omega t)+B\sin(\omega t)=\sqrt{A^2+B^2}\cos(\omega t-\arctan(\frac{B}{A})), \qquad (14)$$

the optical signal term can be written as

$$P_{sig}=2\sqrt{p_S p_R}C_{sig}\cos(2\pi f_S t-\Phi), \qquad (15)$$

where

$$C_{sig}=y_{pmax}\cdot\sqrt{\left(\int_{0}^{x_{pmax}}\cos\left(2\pi f_x x-\frac{\pi x_p^2}{\lambda F}+\phi\right)dx_p\right)^2+\left(\int_{0}^{x_{pmax}}\sin\left(2\pi f_x x-\frac{\pi x_p^2}{\lambda F}+\phi\right)dx_p\right)^2} \quad (16)$$

and

$$\Phi=atan\left[\frac{\int_{0}^{x_{pmax}}\sin\left(2\pi f_x x-\frac{\pi x_p^2}{\lambda F}+\phi\right)dx_p}{\int_{0}^{x_{pmax}}\cos\left(2\pi f_x x-\frac{\pi x_p^2}{\lambda F}+\phi\right)dx_p}\right]. \quad (17)$$

Further application of trigonometric identities allows for Φ to simply be removed from the integrands in Equation (16) for C_{sig}. The final version of the factor which accounts for the finite aperture of the photodetector and multiplies the output current of the photodetector is

$$\alpha=\frac{C_{sig}}{\sqrt{a_S a_R}} \quad (18)$$

where a_S and a_R are the cross-sectional areas of the signal and reference optical beams, respectively.

Hence, the SNR expressed by Equation (8) and calculated in Equation (11) must be multiplied by a factor of α^2. α as a function of true time delay is plotted in Figure 5 for values tabulated in Table 1. The minimum value of α^2 is 0.008, so the corrected value of the SNR, taking into account the effect of a finite aperture detector, is reduced by as much as 20.97 dB, ie,

$$corrected.SNR=10\log(\alpha^2\cdot 8.8x10^7)=-20.97+79.45=58.48dB \quad (19)$$

Hence, depending on the particular spatial frequency, and so time delay and/or RF frequency, a substantial reduction in optical SNR is possible.

IV. RADAR SYSTEM PERFORMANCE PARAMETERS

In this Section, specific radar system parameters - array gain, beamwidth, sidelobe level, and beampattern - will be calculated for the particular photonic beamformer studied here. Comparisons will be made to the ideal conventional beamformer.

A. BEAMPATTERN

The beampattern of an array is a diagram showing the ability of the array to direct energy in any one direction, usually taken as the forward direction (θ_o=0°). Typically, with an array composed of isotropic radiators as elements, energy is only approximately controlled, with much leakage into unintended directions. Control of the direction of emission is improved as more and more elements are excited simultaneously, with the appropriate phases. It is the job of the beamformer to establish the phases of the elemental radiators.

A beamformer with perfect phase shifts for a given radiation frequency, f=ω/2π, and element spacings, d, of one half of the radiation wavelength, will produce the optimal beampattern. Any errors in the phases of the element excitations, such as produced by quantization noise in a digital beamformer, or by the "wiggles" in Φ(x,ω) for the photonic beamformer under consideration here, result in a degradation of the beampattern.

For an ideal true time delay beamformer, steered to angle θ_o and scanned at an angle θ_s, the beampattern is given by [11]

$$b(\theta_o,\theta_s,k)=\left|\sum_{n=1}^{N}\exp\left(j\frac{\omega dn}{c}(\sin\theta_o-\sin\theta_s)\right)\right|^2 \qquad (20)$$

For the photonic beamformer under consideration here, the actual time delay between elements is incorporated into Equation (20) by recognizing that ωdn/c can be replaced by the phase function Φ(x,ω) (Equation (6)), calculated for each specific element n. The ideal and actual beampatterns for a ten-element line array are presented in Figure 6, and will be discussed in more detail in Sections IV.B and IV.C.

B. BEAMWIDTH

The beamwidth is typically defined as the spread of the main lobe of the beampattern, taken between angles at which the power in the pattern is reduced by 3dB. For the ideal conventional beamformer for a ten-element line array, the full beamwidth at half-maximum power (FWHM) is 16.95°, as determined from

the calculations performed to obtain Figure 6. For the photonic true time delay beamformer, as modeled in Equation (6), the FWHM is 16.75°, a value 0.2° (3.49×10^{-3} radians) *narrower* than the ideal convential beamformer. This result is unexpected, and may be a fortuitous phase-related self-focusing due to the non-random nature of the diffraction-induced phase "wiggles" [12].

C. SIDELOBE LEVEL

The manner in which most energy is lost out of the main beam is through projection in the directions of the next highest peaks in the beampattern - the primary sidelobes. The maximum primary sidelobe level for the ten-element line array, for the ideal conventional beamformer, is found to be 13dB down from the mainlobe peak level. The sidelobe reaches its peak value at 17°. Calculations for the same line array using the modeled photonic beamformer yield a peak primary sidelobe level 13.42dB down from the mainlobe peak, *lower* than the ideal conventional beamformer produces. Again, this very slight improvement over the conventional beamformer is unexpected and fortuitous; it may not hold for all array configurations of interest. However, there is certainly no degradation in the case of the modeled photonic beamformer. The sidelobe peak for the photonic beamformer is centered at 16°.

D. ARRAY GAIN

The array gain of a beamformer measures how well the antenna can concentrate energy in a particular direction, compared to an isotropic radiator. It is given by the ratio of the gain achieved in the detection of the coherent signal, over the gain provided by the multi-element array in the detection of the incoherent noise [12]. For an ideal conventional beamformer and a ten-element line array, the array gain is a factor of ten.

The signal gain for a general phased array is given by

$$g_S = \left| \sum_{n=0}^{N} e^{j\omega(\hat{\tau}_n - \tau_n)} \right|^2 = \left| \sum_{n=0}^{N} e^{j(\hat{\Phi}_n - \Phi_n)} \right|^2 \qquad (21)$$

where, for the photonic beamformer, $\hat{\Phi}_n$ is the value provided for the n-th array element by Equation (6). The signal gain for a ten-element line array (N=10), based on the photonic beamformer, is calculated to be 89.375dB. For perfectly incoherent noise, the noise gain is 10dB, and so the array gain is

$$Array.Gain.\ (AG) = \frac{g_s}{N} = 8.9375. \qquad (22)$$

The array efficiency compares the array gain of the beamformer under investigation to that of the equivalent ideal conventional beamformer. Hence,

$$Array.Efficiency = \frac{AG}{10} = 0.89375, \qquad (23)$$

so the photonic beamformer has an array efficiency of 89.375 percent.

V. CONCLUSIONS AND RECOMMENDATIONS

In the work presented here, it was determined that, so far, no major limitations have been identified which would prevent the photonic true time delay line from being an effective beamformer for a radar system. The nonlinearities appearing in the nominally linear time delay versus frequency relationship (the "wiggles"), which were seen experimentally and then analyzed in previous work, do not appear to destroy the capabilities of the delay line.

In fact, the photonic beamformer compares favorably with the ideal conventional beamformer in many performance parameters. The photonic beamformer has a beamwidth and sidelobe level almost identical to those of the standard beamformer. The array efficiency is a respectable 89.375 percent of the ideal.

In the photonic beamformer, the electronic signals generated by the photodetectors at the output of the optical system are appropriately phased RF time-waveforms which can directly drive microwave elements. While the optical signal-to-noise ratio of 58.48dB is good, future investigations must establish if this level is adequate to actually drive microwave elements and still maintain good radar performance.

REFERENCES

[1] Toughlian, E.N. and Zmuda, H., "A Photonic Variable RF Delay Line for Phased Array Antennas", J. Lightwave Technol., vol. 8, no. 12, pp.1824-1828, 1990.

[2] Toughlian, E.N., Zmuda, H., and Kornreich, P., "A Deformable Mirror-Based Optical Beamforming System for Phased Array Antennas", IEEE Photon. Technol. Lett., vol. 2, no. 6, pp. 444-446, 1990.

[3] Zmuda, H. and Toughlian, E.N., "Variable Photonic Delay Line for Phased Array Antennas and RF/Microwave Signal Processing", RL-TR-91-120, Final Technical Report, June 1991.

[4] Zmuda, H. and Toughlian, E.N., "Adaptive Microwave Signal Processing: A Photonic Solution", Microwave Journal, vol. 35, no. 2, 1992.

[5] Sumberg, David A., "An Integrated Photonic Delay Line for Phased Array Antenna Applications", RL-TR-93-54, Final Technical Report, May 1993.

[6] Monsay, Evelyn H., "Photonic Delay Line for High-Frequency Radar Systems", Final Report, Air Force Office of Scientific Research Summer Faculty Research Program, August 1992.

[7] Monsay, E. H., Baldwin, K. C., and Caccuitto, M. J., "Photonic True Time Delay for High-Frequency Phased Array Systems", IEEE Phot. Tech. Lett. Vol. 6, No. 1, January 1994, pgs. 118-120.

[8] Monsay, Evelyn H., "Form and Implications of the Nonlinear Dependence of Phase on Frequency for an Acousto-Optic Beamformer", Final Report, Air Force Office of Scientific Research Summer Faculty Research Program, August 1993.

[9] Private communication from Prof. Henry Zmuda.

[10] VanderLugt, Anthony, Optical Signal Processing, John Wiley & Sons, Inc., New York, N.Y., 1992.

[11] Owsley, Norman L., "Modern Space-Time Signal Processing", Course Notes, Applied Technology Institute, Clarksville, MD, 1993.

[12] Mailloux, Robert J., Phased Array Antenna Handbook, Artech House, Norwood, MA, 1994.

TABLES

FIGURES

TABLE 1

Parameter	Value	Parameter	Value
P_{laser}	35 mW	S	0.76 A/W
T_S	0.5	R_L	50 Ω
T_{PBS}	0.95	i_d	$3x10^{-7}$ A
T_M	0.98	B	200 MHz
T_{AO}	0.16	T	300° K

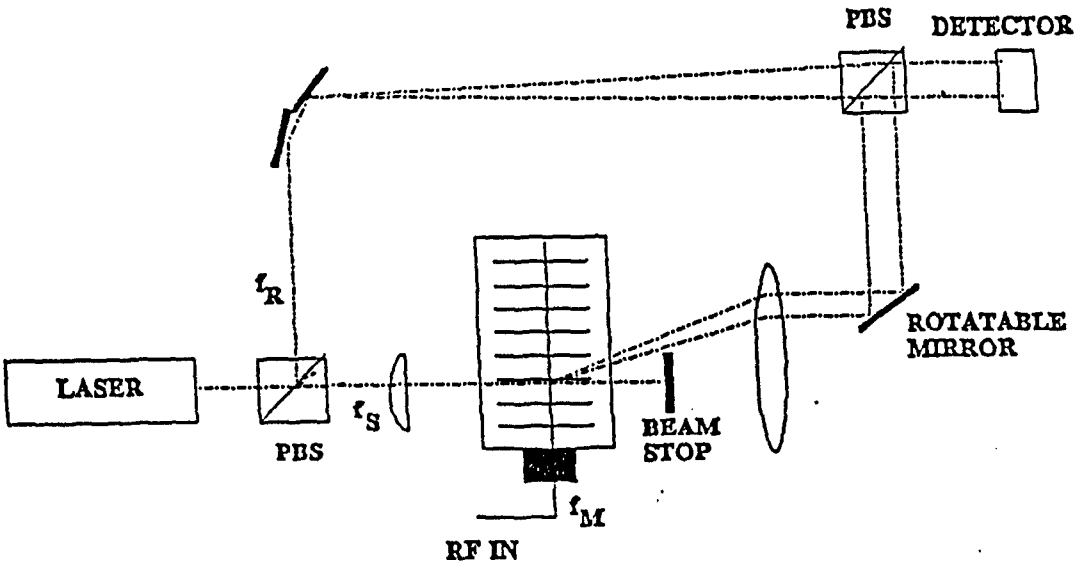


Figure 1

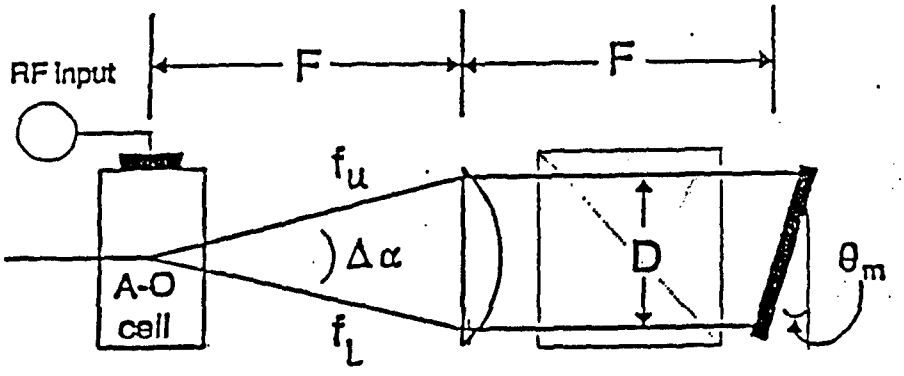


Figure 2

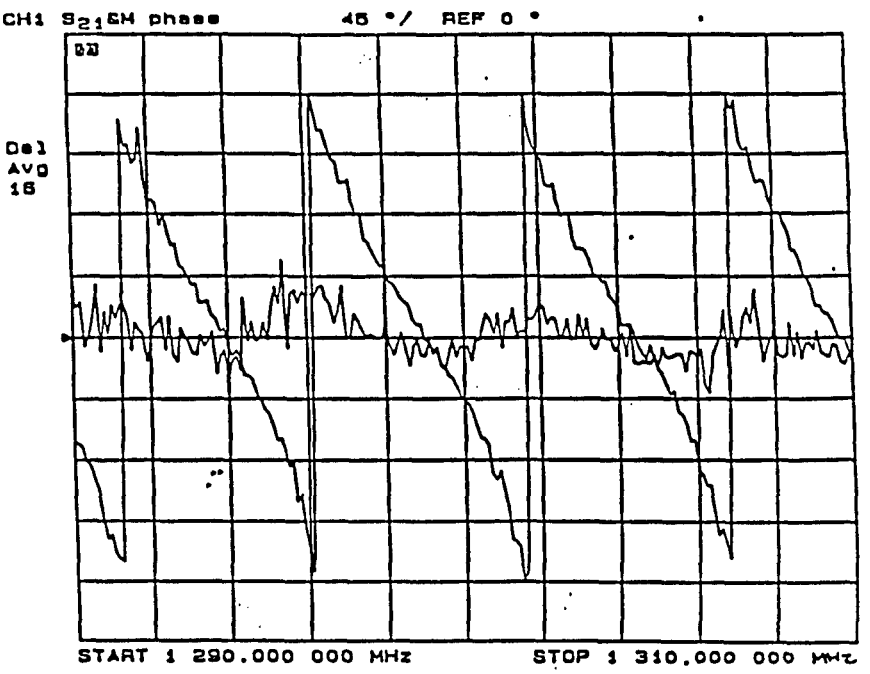


Figure 3

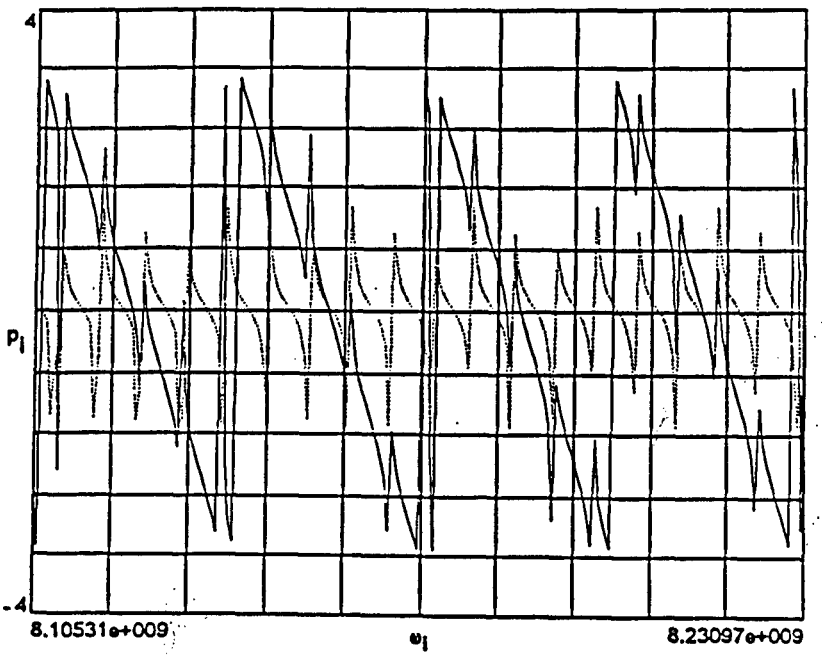


Figure 4

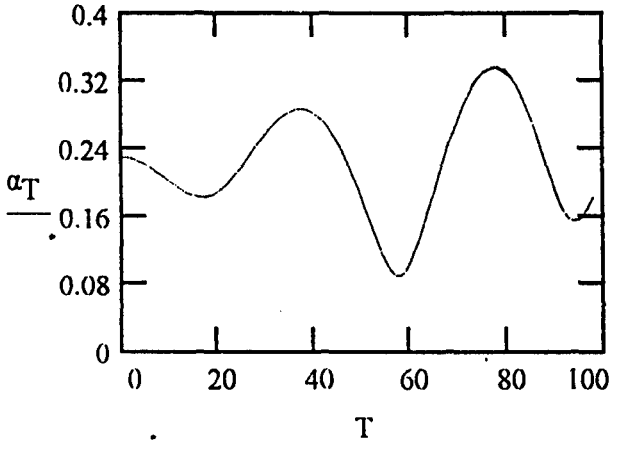


Figure 5

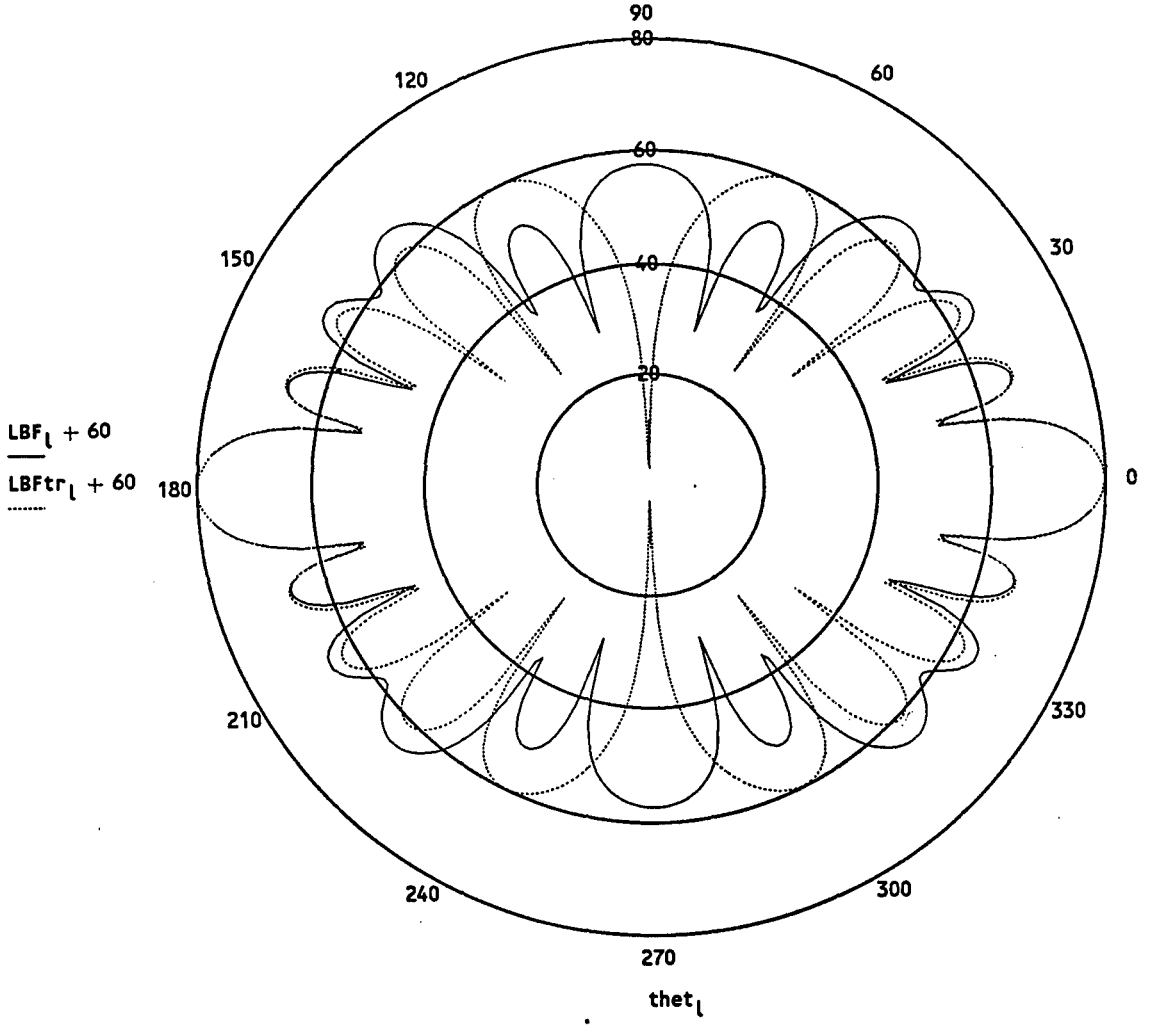


Figure 6

AN EXPLORATORY INVESTIGATION OF MULTIMEDIA REINFORCEMENT FOR LARGE-SCALE INFORMATION RESOURCE MANAGEMENT SYSTEMS: EXPANSION OF THE GUI REPRESENTATION INTO ALTERNATIVE SENSORY REPRESENTATIONS

Michael S. Nilan, Ph.D.
Associate Professor
School of Information Studies
R. David Lankes
Doctoral Student
School of Information Studies

Syracuse University
4-206 Center for Science & Technology
Syracuse, New York 13244-4100

Final Report for:
Research Initiation Program
Rome Laboratories

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, D.C.

and

Syracuse University

December 1993

AN EXPLORATORY INVESTIGATION OF MULTIMEDIA REINFORCEMENT FOR LARGE-SCALE INFORMATION RESOURCE MANAGEMENT SYSTEMS: EXPANSION OF THE GUI REPRESENTATION INTO ALTERNATIVE SENSORY REPRESENTATIONS

Michael S. Nilan
Associate Professor
R. David Lankes
School of Information Studies
Syracuse University

Abstract

With the emergence of powerful desktop computing and the ability to work on multiple tasks simultaneously (multitasking) becoming an everyday reality, representation of diverse processes and data streams becomes more vital. Couple this complexity with the increasingly distributed nature of today's computing and one soon begins to discover the limitation in today's graphical user interfaces (GUIs). This work concentrates on creating a conceptual framework to off-load the complexity of multitasking in a distributed environment and/or in large-scale information resource management systems to alternative sensory representations.

This work, primarily an exploratory work, covers this framework, the creation of an experimental "Sonic Screen" and an empirical test of the effectiveness and efficiency of off-loading certain activities on to aural representations.

AN EXPLORATORY INVESTIGATION OF MULTIMEDIA REINFORCEMENT FOR LARGE-SCALE INFORMATION RESOURCE MANAGEMENT SYSTEMS: EXPANSION OF THE GUI REPRESENTATION INTO ALTERNATIVE SENSORY REPRESENTATIONS

Michael S. Nilan, Ph.D.

R. David Lankes

Introduction

1. Background

The history of computing has been the management of local tasks. Computer representations such as switches, punch cards, text, and later graphics have primarily been geared towards the manipulation and control over files and resources local to a central computer. Even in the days of mainframe dominance user files were stored and processed locally with only dumb terminals geographically distributed. These terminals only provided a window into the "local" environment of the mainframe.

In the 1980's as electronic networking became a larger force in the computing industry, personal computers appeared. These scaled down microcomputers had little or no networking abilities of the larger mainframe and minicomputers. Even the limited telecommunication ability of the modem only allowed a single view of user data stored remotely. This (as it remains today for many users) was a return to the days of a dumb terminal. Users still housed their data in one location and management of user tasks was handled by a single interface. Further, the notion of multitasking, was only represented in a limited fashion, that of "time-slicing." This was where the user was in control of the multiple tasks performed, but was also responsible for switching from task to task and re-representing his/her computing environment.

Personal computers have made great strides in recent history in the area of task and data representation. Possibly the largest advance in the field was the Graphical User Interface (GUI). GUI was pioneered by such visionaries as Eckard, and experimented with at Xerox PARC, but was not mainstreamed until Apple's introduction of the Macintosh personal computer in 1984. Since that time, as more processing power moves towards the desktop, the GUI paradigm has ballooned. Such software as Microsoft Windows, MIT's X-Windows software (and the later incarnations

such as Motif and SUN's Openwindows) have made the use of windows icons and graphics ubiquitous in today's computing. Even non-graphical operating systems such as Microsoft DOS have programs that emulate the GUI interface in certain programs.

These interfaces have dominated the management of users' local workspaces. There is little question about the flattening of the learning curve when employing the windows and icons of the GUI paradigm (for first time computer users at least). User definable file locations, resource allocations and task management have allowed a large number of tasks and or sub tasks to appear to occur simultaneously. By switching between windows, users can control the streams of information needed to accomplish their goals. The GUI interface allows better manipulation of a user's task.

However, the workplace in which the GUI came to prominence is changing. With a new emphasis on bringing sophisticated computer networking to the desktop, the limited local environment is losing dominance. New resources and abilities that exist in other geographic locations and in a vast array of operating environments must now be merged into the user's work place. Users have now lost a great deal of the control over information streams required to complete tasks. The local environment that was shaped by the user must now be mixed with a growing range of resources and data gathered and manipulated anywhere on the globe.

The massive visual bias towards graphics and windows in the GUI paradigm falls short in the new environment. GUI representations soon become cluttered and unmanageable (see figure 1). Icons awash in a sea of color and text bounded by Windows piled upon windows now confuse the user. Few cues aid in the user's navigation of the new distributed work environment. With the exception of an impoverished aural representation, diminished to a single beep of attention which tends to confuse and frustrate users rather than guiding them, other sensory modalities are ignored in favor of visual GUI representations.

This distributed environment is analogous to the emerging large scale information resource management systems where the physical location of the data is almost irrelevant.. The complexity involved in both is immense, and both environments force the user to encounter and control multiple simultaneous data streams. Further, in this new large scale setting (distributed and/or local) the user is no longer able to control the arrangement of files and resource allocations from task to task. It is in the area of managing complex tasks in a distributed environment that this research is directed, specifically, founded on representation of complexity.

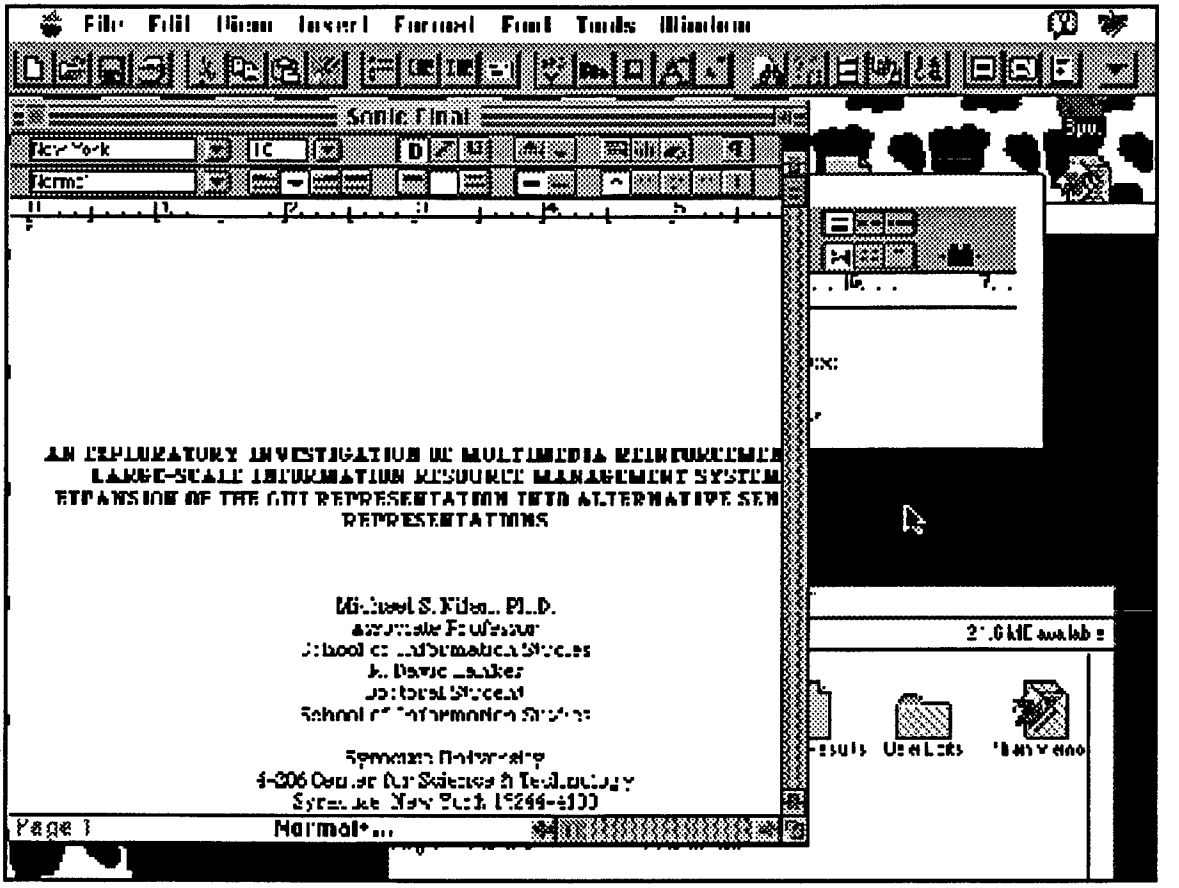


Figure 1: The overcrowded GUI used to produce this document

This research seeks to increase the usability of user/computer interactions in this new desktop environment. We address this by the off-loading of complexity to the aural modality. Sound was chosen for it's present existence in the GUI interface (though minimal) and the ease of implementation using off the shelf existing tools.

We looked to first discover how users use sound. We then sought key activities or tasks in the new distributed workspace. By identifying natural and common uses of aural cues, and matching these to computer functionalities we feel we can increase usability and increase the bandwidth of human/computer interaction while minimizing training. Taken together, increased bandwidth and training reduction, constitute a reduction in complexity.

Discussion of Problem

2. Usability Approach to Representation

The prime motivation behind this study is to increase the usability of the user/computer interface. Usability is defined as the ability of the users to interact with the interface to better accomplish the tasks set before them. In order to achieve this goal one must first understand the two aspects of usability. The first is the way that users naturally solve information problems in everyday life, so called cognitive usability. This problem solving aspect has been covered in other works and can be referenced there (Nilan 1992, "Cognitive Space"). The second aspect of usability is in the way users perceive things, or sensory usability. The goal is to have the user see the system as a natural match between the user's way of solving problems, and the way that the user perceives system representations. This will result in a set of functional requisites that can be implemented in a computer interface. These requisites, for the purpose of this study, are broken into two parts:

- Task functional requisites - These are the key aspects of the new distributed environment and the environment of large scale data management systems. This section seeks to understand what users must accomplish and how distributed resources can fit in with existing local resources. Here we are exploring cognitive usability.

- Aural functional requisites - This is how people perceive via sensory inputs. By better understanding how people utilize the audio modality (in this case), we can better and more naturally implement an expanded aural interface into the system. This is an exploration of sensory usability.

The chief goal of this study is not to introduce new functionalities that must be taught to users, but rather to bring existing functions of users' to problem solving human/computer representation. It is believed that by combining the existent user behaviors into a system, the system will be able to better represent and enrich user/computer interaction. This will increase the users' ability to manage the complexity of multitasking in a large scale information management system. By taking advantage of the richness in human sensory interaction with the environment, we believe we can better match human problem solving and human

sensory perception to increase our chances of improving usability without increasing training.

3. Task functional requisites

To begin to utilize the impoverished audio modality, the research team identified key characteristics of the emerging distributed working environment. The requisites of their environment involved both a strong desktop computing component and an equally important distributed or networked element.

The first task requisite identified was the traditional use of a computer interface, i.e., manipulating local files or resources. This is defined as the functionalities (i.e., what the system can do) and data resident in the desktop computing unit. These resources are under the direct control of the user. Local files are arranged and utilized only in accordance with the user's wants and needs. This definition implies that local files are easily manipulated and the system used for resource allocation and manipulation are representative of a given user's problem solving process. Traditional GUIs and operating systems have been sufficient because they empower users in their personal data organization.

However, as has been stated previously, traditional operating systems become more cumbersome when they encounter distributed data and resources. This data is normally organized by other system entities, and is often arranged for the problem solving needs of others. The user has little, or more often, no control over these remote resources. Network navigation tools and distributed information systems have been developed to explore and manipulate these remote resources, but these systems tend to lack desktop integration and further add to desktop clutter above and beyond the local functions. While allowing for a certain control over distributed resources, the users cannot control the rate of progress, or anticipate the results of the manipulation. For the purpose of task completion, distributed resources remain chaotic.

This lack of resource control forces users to the last of the major task requisites discovered, communication. Conversing with others in a distributed environment either synchronously or asynchronously, is a vital part of the new networked working environment. While done for traditional reasons of recreation and idea exchange, the research team discovered a further need for communication, that of coping with distributed environments. E-mail, currently the most prevalent form of communication in networks, is being used to form collections or groups of

expertise and support with network discovery and literacy. Further, e-mail often is used to contact remote resources when problems are either encountered or anticipated.

So we found three primary task requisites of users in this environment:

- Local resource/file manipulation
- Remote resource/file manipulation
- Communication

In terms of the complexity management focus of this study, all three of these tasks must be performed simultaneously in order to be realistic and must occur in a coordinated manner at the desktop. This type of complexity management burden illustrates the prime focus of this study.

4. Aural functional requisites

While the team looked to the present working environment to determine task needs, this was not a viable approach for determining the uses of sound in a distributed environment. As has been previously stated, the use of sound in present representations is insufficient for the new needs of networked environments. So, we adopted a user based approach (Nilan, "Cognitive Space") to determine how sound was used in everyday life. The objective was to find what users presently use sound for and then match these uses to the task functional requisites listed above. In this manner the use of sound in the resulting representation would seem more natural to the user, decreasing the need for training, and the bandwidth of communication between the users and the computer would be increased due to more robust representation across the human sensory spectrum.

We approached therapists working with the deaf to try and determine how people use audio cues in their day-to-day problem solving process. Our belief being those with full hearing capabilities working with people without such capabilities would be more aware of what hearing is used for. We also approached designers and manufactures of equipment meant to supplement or replace the audio modality in the deaf. Audio cues were found to be integral in determining actions for a desired outcome. Examples ranged from determining the weather, to taking life-saving actions in emergencies.

From the interviews with those working with the deaf, brainstorming, and a series of experiments with differing audio representations we identified the following aural functional requisites:

• Monitoring - Monitoring, an ongoing observation of a given event, can occur in either a continuous fashion or a discrete one. In it's discrete form, it can be called signaling, the instant recognition of an event not across time. Signaling is one of the few audio cues available in GUI interfaces today. The standard system beep is a signal, and therefore an instance of monitoring.

However, it is in the continuous form that we find monitoring under-used in today's computer interface. Human beings use the steady state or variable state of a sound to provide information on a given process in life. This is exemplified by knowing a machine is working by listening for the hum of its component parts.

• Locating - Users often use sound to place objects, event and themselves in space. Often movement and actions are based more on auditory cues than on visual ones. Further, the placement of a given sound often gives a great deal of data when juxtaposed with previous knowledge of the task and situation.

• Channel of Information - Channel of Information is a direct analog of text. It is defined as the transmittal of symbolic information via sound that is translated by a cognitive process as opposed to a physiological one. A telephone conversation would be an exemplar of this use of sound.

Because of its assumed added benefit and the exploratory nature of this study, using sound as a channel of information was not experimentally tested, but would be assumed to be a part of an eventual system.

The research team assumes the effectiveness of this aural function. Since there is a direct relationship between the spoken word and text, any use of speech will increase the bandwidth of communication between the user and the interface. Having the computer read incoming e-mail is an example of this function.

Therefore this study will look at the inclusion of audio functional requisites into a system. The system will then be tested to find added benefit to users in working with distributed or large scale management systems.

Methodology

5. Methods and Experimental Design

This section will present the predictor variables, the tasks, the criterion variables and the basic experimental design.

5.1. Predictor Variables

Once the aural functional requisites were identified, the independent representational variables were determined. These variables were determined for their coverage of the function's attributes and the practical ability to be represented by off the shelf hardware and software. The research team wanted to avoid unique solutions that would be difficult to reproduce elsewhere. However, we wanted a rich enough set of malleable variables that we could explore the impact of expanded aural representations on users.

The following list of variables are assumed to be independent of each other, but not exhaustive of "aural" variables. They were derived from discussions with therapists working with the deaf as the prime variables used in auditory compensation. Further, all predictor variables are assumed to be continuous, but no attempt is made to justify the variable's measurement as more than an ordinal measure unless otherwise stated. Given our heuristic purposes, an ordinal level of measurement was sufficient to demonstrate the applicability of these variables.

- Pitch - Defined as the physical frequency of the sound waves. Commonly referred to as "high" or "low" and equivalent to bass or treble. Pitch is operationalized in this study as a tone of 100 to 1000 Hz.

- Location - Defined as the perceived physical placement of an audio cue. In this study, we restricted these physical locations to either left, center or right channels. While location is a continuous measure, our experimental restrictions operationalized it as a nominal variable with no interplay utilized between the discrete locations.

- Periodicity - Defined as the time between instances of a given audio cue. This period between occurrences can be as small as 0 or infinitely long. Instances were represented as a constant ticking. This variable was operationalized as a changing

frequency of clicks starting at 1 click per second and increasing to a constant tone.

It should be noted that volume is considered a variable as well, but was found to be too dependent on environment and situation. There is too much variability in users' perception of volume to be useful in this exploratory research, but subjects were allowed to adjust volume to their own comfort level.

5.2 Tasks

The experimental tasks were set up in such a way that they mimicked "real world" situations, while allowing the researchers to retain control over the independent variables. The instruments developed covered the three primary tasks defined above: local file manipulation, remote file manipulation, and communication.

This research seeks to explore the expansion of the audio modality to solve real world tasks in a distributed environment. To do this, the research staff sought to find tasks that:

- Accessed local files/resources - such as word processing or desktop publishing
- Accessed distributed or remote files/resources - such as Gopher and other Internet resources.
- Employed remote/local communication - such as e-mail

These functions were chosen for their coverage of the working conditions of a distributed, network environment, and also for the large amount of data previously acquired on the tasks of word processing (Nilan, Newby & Duvall "Word Processing"), networked information sources (Syracuse Campus Wide Information Systems"), and e-mail (Nilan, Travika "User-Defined requirements for electronic mail systems) use.

The tasks decided upon are described in the following sections. The following matrix indicates the way task functional predictor variables were joined with aural predictor variables. Blank cells were not tested, for reasons cited in the footnotes:

Task Functional Predictor Variables by Aural Predicator Variables	**Monitoring**	**Locating**
Pitch	Pitch is used to identify the amount of interaction required by a process, or the closeness of that process to completion. The higher the pitch, the closer an item is to completion, or a need for intervention.	Pitch has nearly the same effect as volume, in that an increase in pitch indicates a closeness. This closeness can either be indicative of completion or physical location.
Location	With the ability to place processes in a given aural location (in stereo left, right or center), the direction from which a monitoring sound is produced is significant.	1
Periodicity	As with pitch, the rate at which an audio cue is repeated indicates the amount of user intervention required. IT also indicates the closeness to completion of a given task.	The faster the rate of an audio signal, the closer a thing is perceived to be. Periodicity acts in much the same manner as pitch and volume in this case.

[1]While this seems like the logical intersection between task and variable, true exploration of this variable is prohibitive due to the cost and effort need to "place" sounds in physical space. Since the present limitation for this study is stereo location, user discretion is utilized for this combination. We would like to try to explore this in the future with a multi-channel audio systems and digital signal processing.

Audio Representations

The following table represents the way in which audio variables were matched with audio functional requisites. See below for actual experimental operationalizations.

	Monitoring	Locating
Pitch	continuous/discrete	continuous/discrete
Location	discrete	discrete
Periodicity	continuous/discrete	continuous/discrete

5.3. Criterion Variables

There were two types of criterion variables used in this study; efficiency and effectiveness. Efficiency is a traditional evaluation criterion variable and is operationalized as the speed at which a user can accomplish their goal. It is measured objectively by the instrument itself. Effectiveness, on the other hand, are measures to gain insight directly from the user about the appropriateness or "fit" between representations and the user's perception of what s/he is doing.

5.3.1. Efficiency

This is a measure of how "fast" a user can accomplish the goal. The question it relates to is simply does the added aural representation allow users to accomplish their tasks more quickly. To measure this we use the following variable:

- Time on Task (TT)- A simple measure in seconds of the amount of time needed to accomplish a given task set before the user (see below for explanations of the tasks used in this research). This time does not include the time needed to record values for the criterion variables.

5.3.2. Effectiveness

These metrics seek to determine the user's reaction to the new aural representations. They also seek to understand why there was or was not a gain in efficiency. Subjects were asked to rate the following variables on a Likert scales from 0 to 10. 0 was defined as "Not at All" and 10 as "Complete."

• Confusion (Cf)- The amount of disorientation felt by the user at a given point in the experiment. The user was asked: *"Please take a moment to rate how confusing you feel the system is."*

• Confidence (Cn)- The perceived ability of the user to confidently complete the task and or move forward in his or her problem solving exercise. The user was asked: *"Please assess how confident you felt about your ability to complete this task using this tool."*

• System Knowledge (Sk)- The perception that the user knows what the system as a whole is doing. The user was asked: *"Did you feel you knew what the system was doing during this task?"*

• Program knowledge (Pk)- The perception of the user that the s/he is aware of the programs presently in use on an individual basis. The user was asked: *"How aware of the programs activities were you?"*

5.4. Experimental Design

In the following section a task is defined as the overall goal of the user. In the case of this experiment, the completion of a newsletter. A sub-task is then defined as a finite process needed to be accomplished to forward a user towards his or her goal. So a task is composed of several sub-tasks.

Each task allowed the subject to use any application at any time during that sub-task (the tasks had to be completed in order). They were given a sub-task to accomplish, but were be asked to do auxiliary functions during the time it takes for applications to complete their processes (they, for example, edited a previous block of text while waiting for e-mail to arrive). The total time needed to complete a sub-task was timed. At the end of each sub-task the subject was asked to comment on the criterion measures. The subject was allowed as much time as desired to evaluate these measures (Likert scales). At the end of the experiment the subject was asked for demographic information as well (see below).

5.4.1. Demographic Information Compiled

USERID - Number assigned by researcher to track results

Name - Name of Subject

Age - Age of Subject

Number of years computer experience - Number of years using a computer. This was an attempt to determine comfort level and experience with computers in general

Level of expertise - Measured by the nominal levels of "novice," "intermediate," or "expert." The selection of these categories were self-selected by the subjects.

Platform experience - Measured by the nominal levels of "IBM (DOS/Windows)," "Macintosh," "UNIX," or "Other." Once again a means of determining if the results were due to the system, or the prior experience of the subjects.

Auditory Problem - Measured by the nominal levels of "yes," or "no." The user was asked "Are you in any way, or have you ever been diagnosed with a chronic auditory problem?"

Program Familiarity - The subject was asked to answer "yes" or "no" to whether they had knowledge of the following programs:

HyperCard: This was the development platform of the experimental system. We wanted to know if subjects were going to pay more attention to the development of the systems than the system itself.

NCSA Mosaic: A mock-up of Mosaic was used in the experiment. The Mock-up closely simulated the operation of Mosaic, but was different enough (for purposes of data collection and the experimental setting) that users might be confused.

POPMail: A mock-up of POPMail was used in the experiment. The Mock-up closely simulated the operation of POPMail, but was different enough that users might be confused.

5.4.2. Experimental Tasks

The problem (task) chosen was that of composing a newsletter that integrated sources from local sources and remote sources. The newsletter consisted of several parts (see Figure 2). These sub-tasks were, in order:

- Accessing a local file
- Using e-mail to compile an interview;
- Finding and retrieving a remote graphic
- Finding and retrieving a remote text

- Utilizing a local database
- Finding and utilizing a remote database.

Each of these "sub-tasks" were accomplished at the same time as editing placed text from previous sub-tasks.

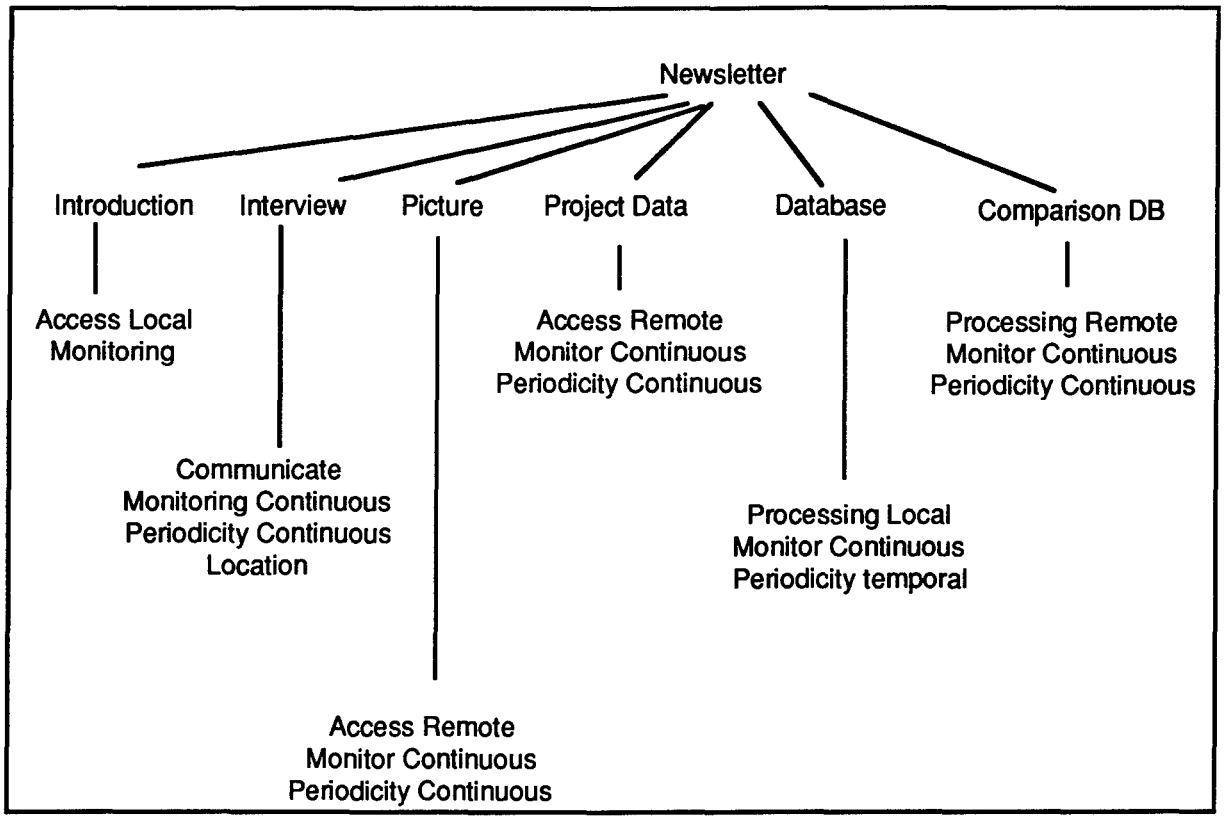


Figure 2: Composition of the Newsletter

In the following experimental implementations, these conventions are used throughout. Each item below indicates a separate sub-task in the experimental problem:

- Introduction - Placement of a text from the "Finder" application. This application resembled the look and functionality of the Macintosh Finder application in which users navigate through folders to find local files and applications.

- Interview - The "POPMail" application, a simple e-mail program based on the University of Minnesota's e-mail application of the same name, was used. The

subjects had to compose and send an e-mail asking for the question and the answer to the interview. Sixty seconds after the e-mail was sent the question arrived, sixty seconds after that the answer arrived.

• Picture - A mock-up of "NCSA Mosaic," a distributed multimedia hypertext application, was used. Subjects had to navigate through hypertext links, find the picture, and then wait thirty seconds as the file was retrieved. The users then placed the graphic on the page.

• Project Data - The "Mosaic" application was used again to locate and retrieve text on a project related to the interview.

• Local Database - A "Database" application was used to compile a statistic. The compilation took approximately forty seconds to accomplish. The statistic was then placed into the newsletter.

• Remote Database - The "Mosaic" application was used to find and activate a remote database by clicking on a menu item. The results were then placed into the newsletter.

The following are the experimental representations used in accomplishing the above sub-tasks:

Group 1 - The control group. These subjects were given no sonic representations to assist them in performing their sub-tasks. They were given only visual cues matching those of the Macintosh GUI on which the experimental system was developed.

Group 2 - This group used pitches to represent all aspects of sub-task function as a compliment to the visual representation. For example, when the database was processing its results, the subject heard a pitch that climbed as the process was nearing completion. This was used as the primary experimental group.
• Introduction - A continuous pitch rose as users moved closer to a given file, and the pitch dropped as users moved farther away from the file.

• Interview - As e-mail was sent, two pitches were used to represent the two e-mail messages (the question and the answer) pending. Each continuous tone was located in a separate discrete audio channel (the question in the left ear and the answer in the right ear). As an e-mail arrived, the tone in the appropriate location stopped.

• Picture - A tone (located in the center channel) was started as subjects entered the "Mosaic" application. As the users navigated through the hypertext links to files that were located geographically farther away (therefore taking longer to retrieve files) the pitch of the tone dropped. As the picture was located, and selected for retrieval the pitch began to rise. When it reached the original pitch present when entering the Mosaic application, the picture appeared for placing.

• Project Data - Same representation as above. However, in this case there was little time delay in retrieving the text.

• Local Database - Once the statistic was being compiled, a tone began to continuously rise in pitch. As the pitch rose to a certain level, it stopped indicating the process was complete.

• Remote Database - Same representation as the Picture sub-task. Instead of the picture appearing for placement, the statistic appeared in the Mosaic application.

Group 3 - This group used a series of ticks to represent all aspects of sub-task functions in compliment to the visual representation. For example, when the database was processing it's results, the subject heard a series of ticks that continuously increased in speed as the process was nearing completion. This was primarily a test of periodicity. The researchers wanted to find out if tone and periodicity were interchangeable.

• Introduction - A ticking grew faster as users moved closer to a given file, and the speed dropped in a continuous manner as users moved farther away from the file.

• Interview - As e-mail was sent two consistent rates of ticks were used to represent the two e-mail messages (the question and the answer) pending. Each series of ticks was located in a separate audio channel (the question in

the left ear and the answer in the right ear). As an e-mail arrived, the ticking in the appropriate location stopped.

• Picture - A series of ticks (located in the center channel) was started as subjects entered the "Mosaic" application. As the users navigated through the hypertext links to files that were located geographically farther away (therefore taking longer to retrieve files), the rate of the ticking dropped. As the picture was located and selected for retrieval, the rate began to rise. When it reached the original rate present when entering the Mosaic application, the picture appeared for placing.

• Project Data - Same representation as above. However, in this case there was little time delay in retrieving the text.

• Local Database - Once the statistic was being compiled, the rate of ticks increased. As the rate rose to a certain level, it stopped indicating the process was complete.

• Remote Database - The rate of ticking was dropped as subjects navigated to more distant resources. As they began the remote statistic compilation, a pitch was used to represent the progress of the remote database. The tone rose as the compilation neared completion.

At the end of each of these sub-tasks effectiveness measures were obtained.

5.5. The Sample

There were 14 subjects used in this exploratory experiment. All reported having roughly equivalent computer backgrounds and experience levels. While there were variations in years of computer knowledge, this did not seems to effect the results of this experiment.

The subjects were all taken from a single undergraduate level course that strongly emphasized computer based skills. All had the same background on the Macintosh computer (used for delivery of instrument). It is the belief of the researchers that the variations in the results of the experiment are due to the experimental treatments, not previous computer knowledge.

Results

6. Results

Initial exploration of the results show an improvement in efficiency when sound is used. This seems to be related to the user's knowledge of program activities. The control group performed the tasks in a serial manner, switching between tasks

only when a task (such as editing) was completed. The experimental groups, however, coordinated the tasks, giving functions attention when needed.

Effectiveness showed marginal improvements between groups. Observations of the subjects during the experiment indicated a greater improvement than is indicated by the subjects' responses. This trend in responses seems strongly related to the experience level of the user. The more experience with computers in general, the more critical a user was when determining his or her criterion measures.

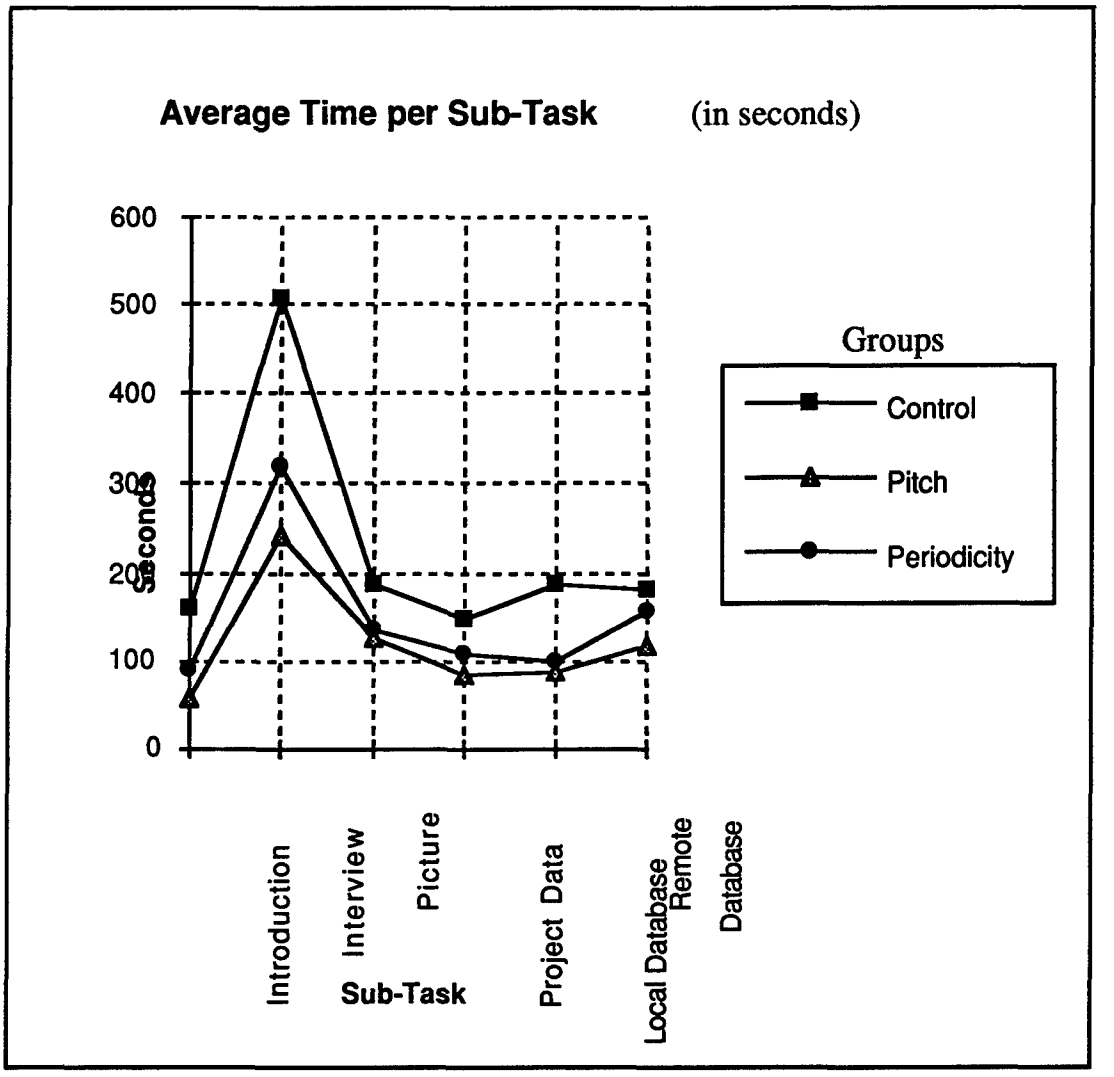
As a whole the alternative representation utilized in this work clearly demonstrates a need for more study. However, with the results gathered, there is a clear benefit to be gained by the off-loading of complexity in to the aural modality.

6.1. Results by Mean scores

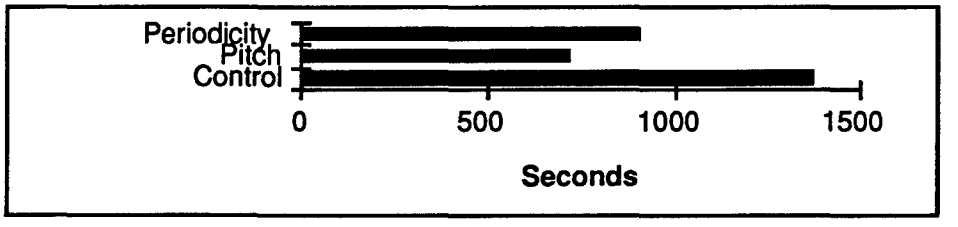
Break Down of Sample by Experiment:

Experiment (Prime Characteristic)	Number of Subjects
Group 1 (Control)	5
Group 2 (Pitch)	5
Group 3 (Periodicity)	4

The Following Charts are Means of the Given Group Scores:

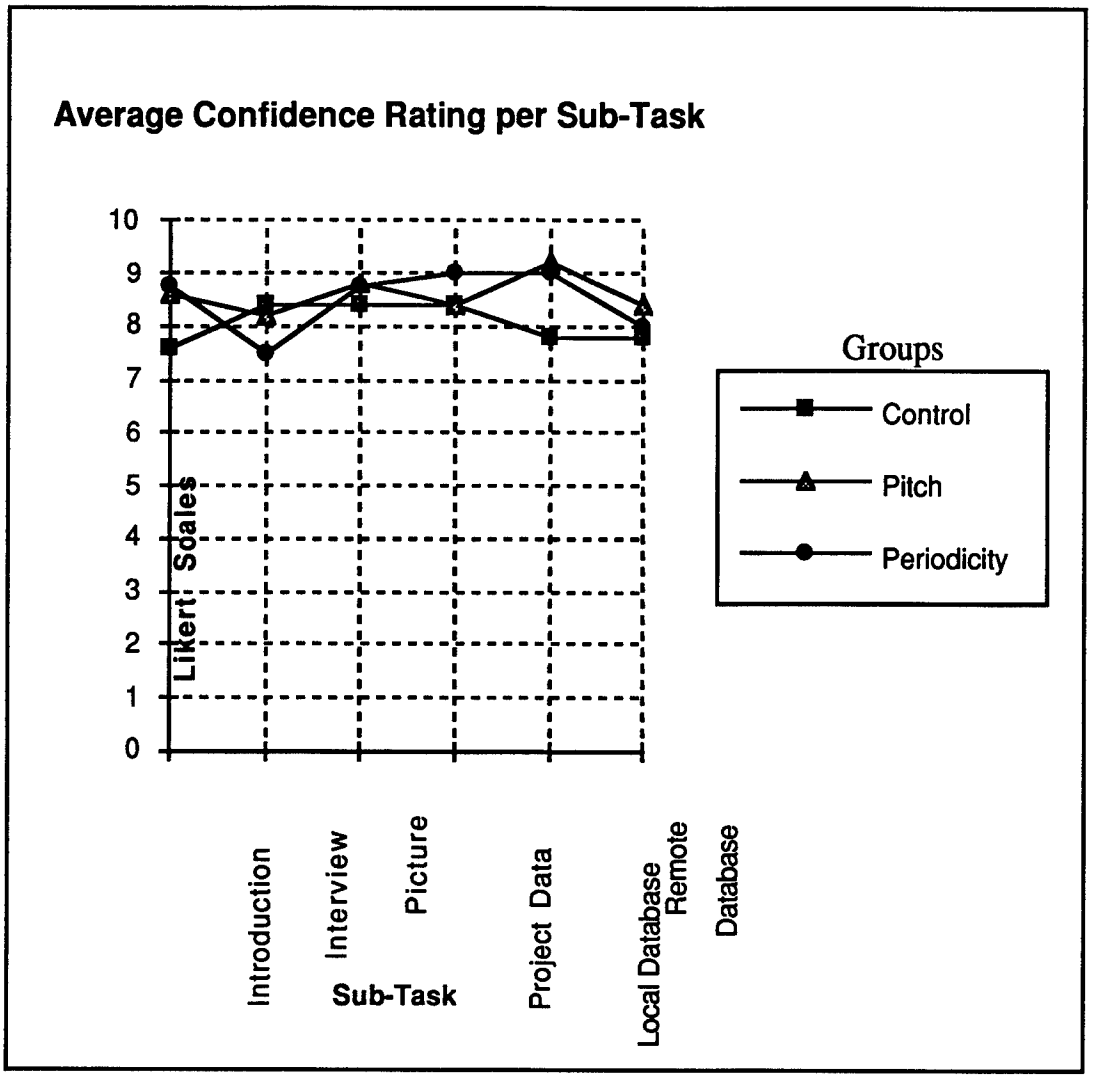


	Intro	Inter	Picture	Project Data	Local DB	Remote DB	Total Time in Seconds
Control	161	506.8	188.4	149.2	189.4	180.8	1375.6
Pitch	58.6	242.4	128.4	85.2	88.8	119.4	722.8
Periodicity	90.75	317.25	136	108.25	99.75	159	911



The lower the number the better the representation.

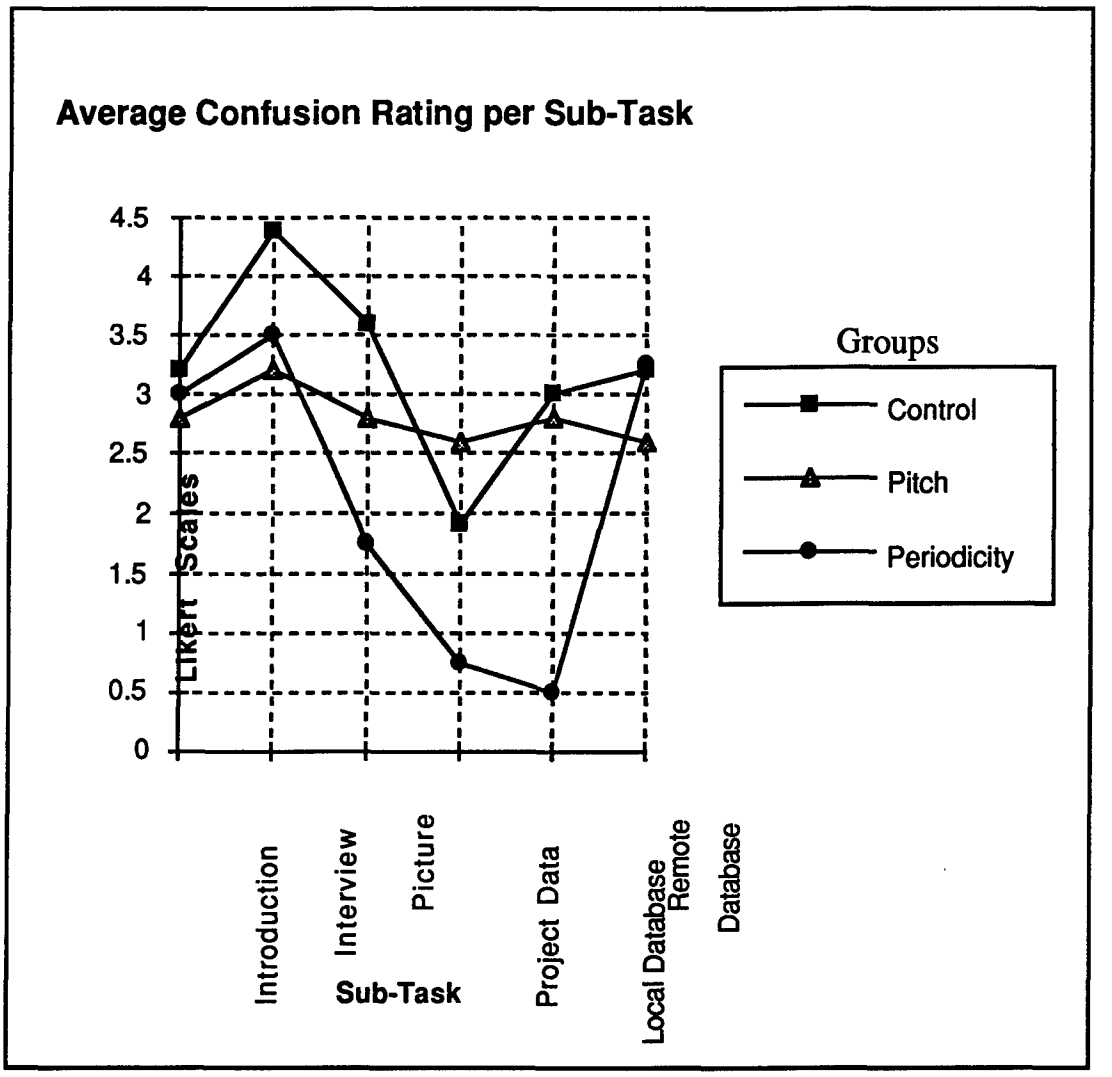
We see in the above chart that the control group took the longest to complete all sub-tasks. Both experimental groups were faster, with the "pitch" group being marginally more efficient that the periodicity group.



	Intro	Inter	Picture	Project Data	Local DB	Remote DB	Average
Control	7.6	8.4	8.4	8.4	7.8	7.8	8.067
Pitch	8.6	8.2	8.8	8.4	9.2	8.4	8.6
Periodicity	8.75	7.5	8.75	9	9	8	8.5

The higher the number the better the representation.

While overall the experimental groups showed higher confidence, the above chart indicates no clear results.

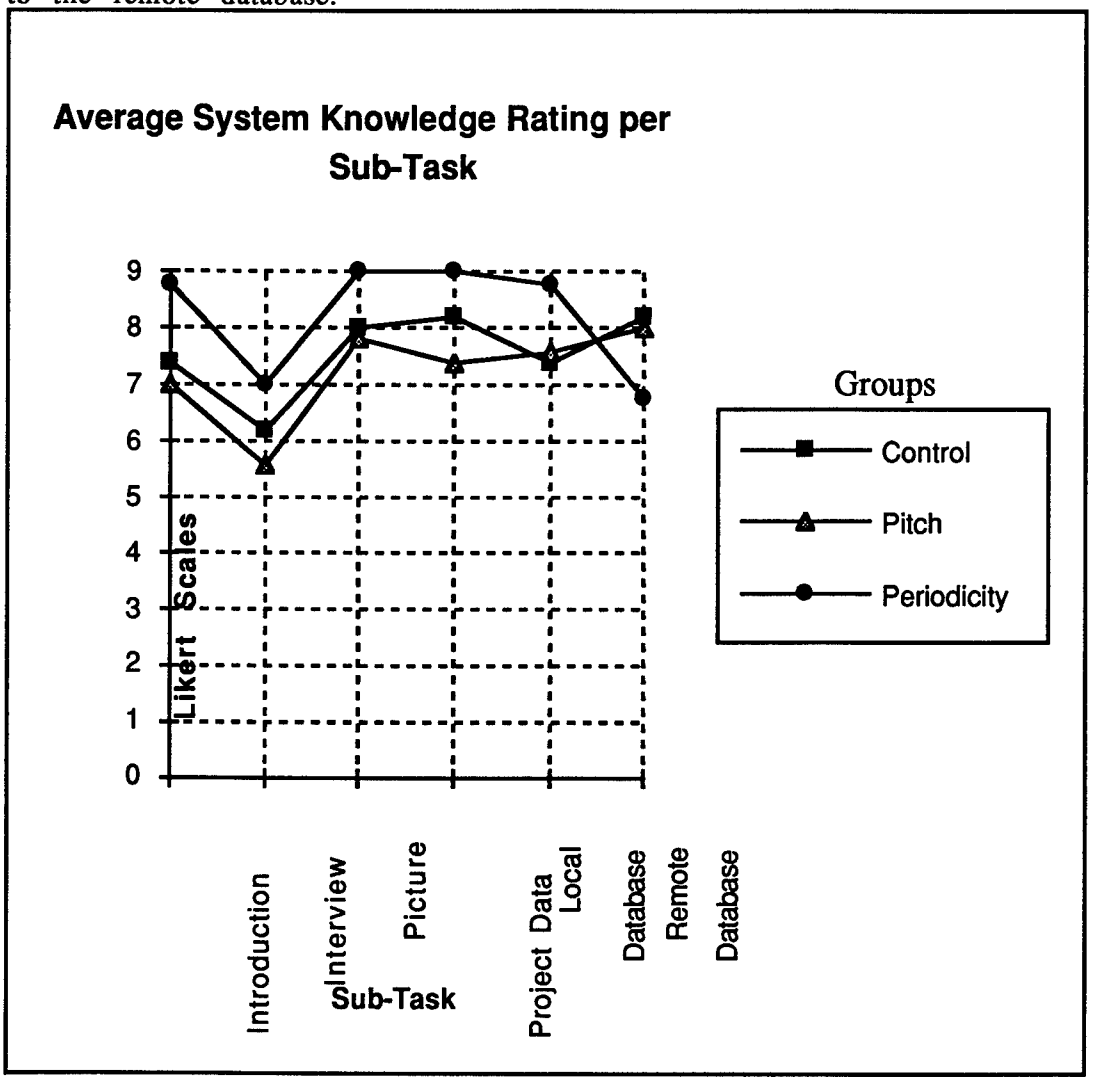


	Intro	Inter	Picture	Project Data	Local DB	Remote DB	Average
Control	3.2	4.4	3.6	1.9	3	3.2	3.2
Pitch	2.8	3.2	2.8	2.6	2.8	2.6	2.8
Periodicity	3	3.5	1.75	0.75	0.5	3.25	2.1

The **lower** the number the better the representation.

We see some interesting trends when looking at confusion ratings (here put on a 0-4.5 scale for emphasis). Overall both experimental groups showed less confusion with the system. The interesting trend, however, is the drop and then subsequent raise in confusion scores as seen above. The initial drop is most likely due to a growing "comfort level" with the interface; as users worked with the instrument, their confusion decreased. The final rise in the control group comes during the use

of local tools, but these tools coincide with text editing. It is likely the seeming lack of feedback from the system during editing periods is what caused increased confusion. With the periodicity group it was observed the confusion was caused by the interface to the remote database.

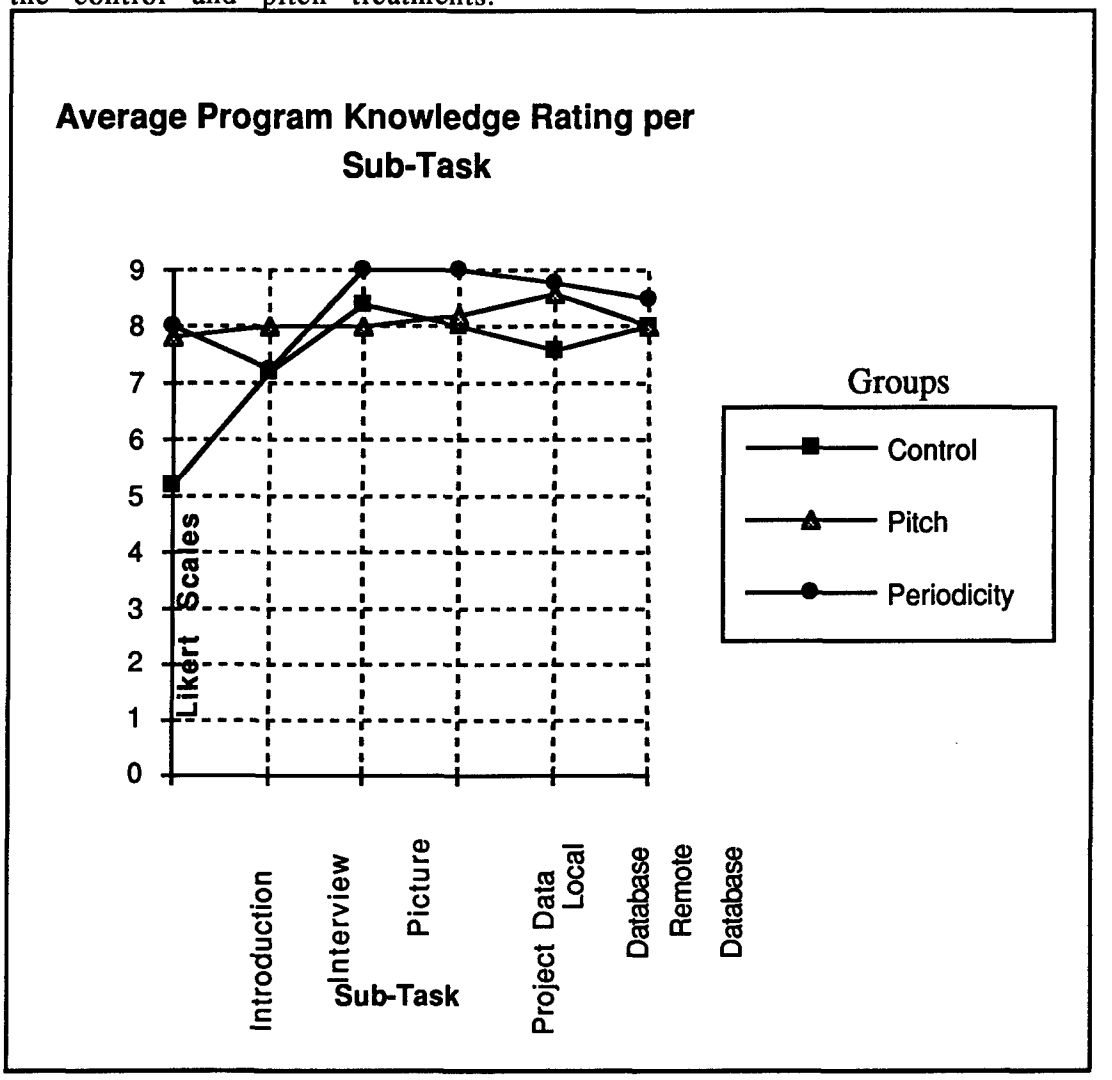


	Intro	Inter	Picture	Project Data	Local DB	Remote DB	Average
Control	7.4	6.2	8	8.2	7.4	8.2	7.6
Pitch	7	5.6	7.8	7.4	7.6	8	7.2
Periodicity	8.75	7	9	9	8.75	6.75	8.2

The higher the number the better the representation.

We see above (on a 0-9 point scale for emphasis) an interesting reversal at the last sub-task. This is most likely due to the change in representation from periodicity

to pitch for the final sub-task (see section 5.4.2.). Periodicity performed the best overall, and for the first five sub-tasks. There seems to be no clear results between the control and pitch treatments.



	Intro	Inter	Picture	Project Data	Local DB	Remote DB	Average
Control	5.2	7.2	8.4	8	7.6	8	7.4
Pitch	7.8	8	8	8.2	8.6	8	8.1
Periodicity	8	7.25	9	9	8.75	8.5	8.4

The higher the number the better the representation.

In this variable we see both experimental groups gaining a higher overall score. Aside from one sub-task pitch outscored the control group tying on one sub-task (Remote DB).

6.2. Summary of Results

- There was a 47% reduction in time needed to complete the overall task using pitch compared to control (34% using periodicity).

- For local tasks (placing the Introduction, and utilizing the Local Database) pitch scored higher in confidence and program knowledge, and lower in confusion.

- For the Interview, where location was utilized, both auditory treatments scored higher in confidence, system knowledge and program knowledge while creating lower confusion.

- In the confusion chart on page 7-23 one sees a decrease in confusion for pitch over time. This seems to indicate a level of comfort with the new audio functionalities over time.

6.3. Other Findings:

These findings, while not central to the experiment, effected in the design of the prototype system. The findings are presented in a general context, and seem valid issues for further study:

1. Users are unable to distinguish between tones that are close in pitch. Tones with similar frequencies are "blended" by the user into a single tone even when separated into a right and left channel.

Pitches with large variation in their frequencies are distinguishable. This study was more concerned with the operationalization of representation than in an empirical study of the effects of pitch upon the physiological function of the auditory system, so a more precise range was not identified.

2. Users perceive two sonic cues with similar frequencies when one of the cues is modulated (has a discernible periodicity). The users, in essence, detect the constant change in state of one sound versus the other.

3. Signaling is valuable in determining the completion of a task, but a constant monitoring tone seems to allow users more knowledge of the programs' workings so they can better coordinate these programs' outputs.

Conclusions

7. Conclusions

This paper has presented a way of thinking about alternative representations in large-scale information resource management systems. The auditory modality was used due to the ease in which it could be implemented. However, now that the conceptual framework has been initiated, further research can explore other sensory modalities for such off-loading of complexity. In fact, the auditory modality can be further implemented using four channel sound to increase the "Sonic Screen" from a singular, linear dimension to a planer representation. One can imagine "locating" processes anywhere on a clock face..."put this e-mail at the 4 O'clock position..."

It has been demonstrated by not only conceptual argument, but also by limited empirical experimentation, that multi-sensory off-loading is a useful approach for managing complexity. Efficiency is increased by adding natural auditory functions to existing user/computer interaction. This efficiency will be crucial as systems increase their scope and complexity. Today's GUI is not robust enough to handle the massive amount of data being processed in an information rich distributed world.

The experiment in this study yielded useful results. It clearly indicated the significant gain to be had by matching users' audio functional requisites and computer representations. However, it was somewhat limited in its ability to implement effectiveness variables. The experimental system needs to be expanded and subjects given more time to familiarize themselves with the system. A longitudinal study would yield richer data about the long term effectiveness of this approach. We would also like to increase the subjects tested.

7.1. Recommendations for Further Research

This study was exploratory in nature and certainly did not seek to answer the full range of questions this research suggests. It did attempt to define a process of investigating the effects of off-loading visual congestion on to alternate sensory modalities. Further study is need to address the problem of initial user confusion in the experiment (novelty of the interface), alternate sensory modalities (what if

sound is not used, but touch), and the creation of an expanded set of criterion variables for determining the effectiveness of interface representations. We recommend the following studies:

- A longitudinal study with a system like our experimental instrument. Does familiarity with the experimental system involving alternate auditory representation increase efficiency and effectiveness over time?

- Further exploration of variations among our predictor and criterion variables.

References

Nilan, Michael S. and Travica B.: User-Defined requirements for electronic mail systems. Report filed with the CASE Center, Syracuse University, 1992.

Nilan, Michael S.:"Cognitive space: A human communication approach to virtual reality for large information resource management problems" by "Forthcoming in JOURNAL OF COMMUNICATIONS, Special Issue on Virtual Reality": Syracuse Univerisity, New York 1992.

Silverstein, Joanne and Nilan, Michael S. "Syracuse University Campus-Wide Information System": CASE Center Technical Report No. 9314. June 1993.

Nilan, Michael S. and Newby, Greg and Duvall, L.: "A User based approach to word processing and office automation.": Syracuse University, New York 1991.

SUPPORTING SYSTEMATIC TESTING FOR REUSABLE SOFTWARE COMPONENTS

Allen S. Parrish
Assistant Professor
Department of Computer Science

The University of Alabama
Box 870290
Tuscaloosa, AL 35487-0290

Final Report for:
Research Initiation Program
Rome Laboratory

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, D.C.
and
The University of Alabama

December 1993

SUPPORTING SYSTEMATIC TESTING FOR REUSABLE SOFTWARE COMPONENTS

Allen S. Parrish
Assistant Professor
Department of Computer Science
The University of Alabama

Abstract

Rome Laboratory is currently developing a certification framework for reusable software components. Such a certification framework must necessarily involve functional evaluation of component correctness. Because of the practical problems with formal verification, this functional evaluation normally includes dynamic testing, as well as static analysis activities. To facilitate the development of a quantitative certification framework involving testing, the use of systematic, measurable testing strategies is necessary. The work reported here involves the development of a tool to support the application of systematic testing strategies to reusable software components. Such a tool could be eventually integrated with the Rome Laboratory certification framework.

SUPPORTING SYSTEMATIC TESTING FOR REUSABLE SOFTWARE COMPONENTS

Allen S. Parrish

1 Introduction

Rome Laboratory [16] is currently developing a certification framework for reusable software components. Such a certification framework must necessarily involve functional evaluation of component correctness. Because of the practical problems with formal verification, this functional evaluation normally includes dynamic testing, as well as static analysis activities. To facilitate the development of a quantitative certification framework involving testing, the use of systematic, measurable testing strategies is necessary (*e.g.*, white-box strategies, such as branch coverage).

A substantial part of the success in current reuse practice involves modules that export *abstract data types*. An *abstract data type (ADT)* consists of a user-defined type and its associated operations, where the representation of the type is kept hidden by the exporting module. [1] Ada packages exporting ADTs are prevalent in the ASSET [1] and RAPID [15] reuse libraries (*e.g.*, the well-known Booch data structure components [4]). ADT modules are the fundamental modularization construct when using the object-based and object-oriented development paradigms. With the increasing popularity of these paradigms, the proportion of such components in reuse libraries is only likely to increase in the future. Consequently, any certification framework for reusable modules must be capable of handling ADT modules.

To implement the Rome Laboratory reuse certification framework, it is therefore important to be able to apply systematic, measurable testing strategies to ADT modules. However, ADT modules are *passive* [8], in that they cannot be independently executed. Consequently, it is impossible to test an ADT module *at all*, without some type of test driver to execute the module. There is no reason to expect a test driver to exist for a particular module in a reuse library, since the module may be developed without an application in mind. Further, the manual development of test drivers is a tedious and error-prone process.

In this work, our major goal was to develop a tool to support the automatic generation of test drivers for reusable software components written in Ada. Ada was chosen as the language for the tool, in that much of the software within government reuse libraries is developed in Ada [1, 15]. In order to support the Rome Laboratory reuse certification framework, drivers are generated that permit *systematic* testing criteria to be applied. This is accomplished by permitting the user to choose between two different testing modes. The first mode allows the user to choose individual operations to test; systematic testing is accomplished by applying branch coverage tools (such as the ATVS [2] tool developed by Rome Laboratory) to measure the degree of coverage of the individual operations.

[1] The classic ADT example is that of *stacks*, whose operations normally include `PUSH`, `POP`, `IS_EMPTY`, etc. The type representation for *stack* (*e.g.*, array with top pointer) is normally hidden from the module client.

The second mode generates a driver that executes a systematically-chosen sequence of operations, based on criteria proposed by Zweben, *et al.* [20]. In this report, we refer to techniques for selecting operation sequences as *sequence testing* techniques.

Our work specifically falls into two general areas:

- Development of a general theory for ADT sequence testing techniques. This theoretical work was published in a high-visibility journal [11], and presented at several conferences [12, 13, 14]. This theory should provide a reasonable foundation for future work in this area.
- Development of a windows-based, functional prototype of the driver generator tool. Our emphasis was on developing a functional, easy-to-use version of the tool, in order to provide a basis for future use and experimentation.

The remainder of this report is organized around these two issues. We first consider our sequence testing theory in Sections 2 and 3. Section 2 provides the background for understanding the original criteria as proposed by Zweben, *et al.* [20]. Section 3 discusses our theoretical contribution, which falls into two categories: (a) clarifying the definitions of the testing criteria; and (b) providing a collection of theorems regarding the implementability of the criteria. Lastly, Section 4 provides basic information regarding our prototype driver generator tool. Note that an executable version of the tool (IBM-PC/MS-Windows version) will be provided directly to Rome Laboratory.

2 Background

2.1 Basic Definitions

Since our interesting is in testing ADT modules, we first define a set of standard terms used in this discussion relative to object-based and object-oriented development. Our definitions of these terms are similar to those used in the object-oriented testing discussion in [5]. We use the term *class* to refer to the implementation of an abstract data type within object-oriented and object-based languages (such as Ada). An *object* is an instance of a class. Instances are created by invoking a special constructor operation (such as `NEW`) on a variable whose type is that of the given class. A class implementation is defined in two parts: an *interface* consisting of a list of operations that can be performed on instances of the class, and a *body* consisting of implementations of the operations. Operations are sometimes called *methods*, and invoking a method with respect to a given object is sometimes referred to as "sending a message" to the object. (In the remainder of this paper, we use these two forms of the terminology interchangeably.)

We assume that it is plausible for a formal, functional *specification* to be associated with a class. The specification defines correct behavior for the class. In practice, specifications are frequently informal (or nonexistent). However, classes can also be specified using formal techniques, and much of the research work involving formal specification techniques has been related to specifying classes.

There are two formal specification techniques commonly discussed in the literature: *algebraic* specifications [9] and *model-based* specifications [17]. Weide, *et al.* [18] contains an excellent overview of these specification techniques. An algebraic specification defines the behavior of a module in terms of a set of axioms that characterize the equivalence of combinations of operations, while model-based specifications involve the individual modeling (using well-defined concepts) of each operation in the class. Table 1 presents both algebraic and model-based specifications for a stack class (similar to [18]).

Formal Module Specifications

Algebraic	**Model Based**

module STACK_TEMPLATE(type T)
  type STACK
  functions
  NEW: → STACK
  PUSH: STACK × T → STACK
  POP: STACK → STACK
  TOP: STACK → T
  domain conditions
  POP(s): not (s = NEW)
  TOP(s): not (s = NEW)
  axioms
  (1) not (PUSH(s, x) = NEW)
  (2) POP(PUSH(s,x)) = s
  (3) TOP(PUSH(s,x)) = x
end STACK_TEMPLATE
	module STACK_TEMPLATE(type T)
	  type STACK is modeled by STRING
	  interface
	  operation New (s: STACK)
	  ensures s = *e*
	  operation Push (s: STACK, x: T)
	  ensures (s = #x ∘ #s) ∧ (x = #x)
	  operation Pop (s: STACK)
	  requires not (s = *e*)
	  ensures ∃ x ∋ #s = x ∘ s
	  operation Top (s: STACK, x: T)
	  requires not (s = *e*)
	  ensures (∃ t ∋ s = x ∘ t) ∧ (s = #s)
	end STACK_TEMPLATE

Table 1

Considering first the algebraic specification, we note the specification is composed of three separate elements: functions, domain conditions, and axioms. The '*functions*' section defines signatures of the class operations. The '*domain conditions*' permit restrictions to be placed on the input values for a particular operation. For values not satisfying the domain conditions, the output of the operation is unspecified. Finally, the '*axioms*' listed in the specification must be satisfied by the implementation. For example, axiom (2) states that the state resulting from executing the sequence `POP(PUSH(s,x))` must be identical to the original state *s* of the stack.

For the model-based specification, we note that each operation identifies both a '*requires*' and an '*ensures*' clause. The '*requires*' clause asserts any domain restrictions on the operation, similar to the 'domain conditions' section of the algebraic specification. The absence of a requires clause implies there are no domain restrictions. The '*ensures*' clause indicates a specific condition that holds true after invoking the operation, provided that the requires clause is met.

With model-based specifications, an independently defined model is chosen to represent the

class. In this case, a string is used to model a stack. Since strings are a well-defined concept from mathematics, the behavior of stack operations can be expressed in terms of string theory operations. For example, e refers to the empty string; the operation *new* returns a empty stack, which is modeled as an empty string. The requires clause for *pop* demands that the stacks provided as input are non-empty; given that this restriction is met, the ensures clause indicates that the new stack s is equal to some element concatenated (denoted by the operator $\circ$) with the old value of the stack (denoted using #s) . Concatenation is used to model this operation as it is a well-defined operation within string theory.

2.2 Conventional Flowgraph-Based Testing

In order to understand flowgraph-based testing techniques for classes, we first review flowgraph-based testing techniques for conventional program modules. Conventional program modules can be decomposed into *blocks* of statements. A *block* is a sequence of statements having the property that it can only be entered through the first statement. Once the first statement is executed, all remaining statements are executed in order. No statement can be added to the beginning or end of a block and still retain the property that whenever the first statement is executed, the rest of the statements are executed in order. In our flowgraph model, a node represents a block, and an edge between two nodes represents the possibility that control is permitted to flow in the direction indicated by the edge. If a transfer between blocks is based on some condition (*e.g.*, from a predicate in an *if* statement), then the edge corresponding to that transfer is *labeled* with the condition.

Techniques (sometimes called *criteria*) used for flowgraph-based testing of conventional programs are typically classified as either control-flow criteria or data-flow criteria. The criteria for which we will discuss class-based analogs are shown in Table 2 below.

Conventional Flowgraph-Based Testing Criteria	
Control Flow	**Data Flow**
node coverage branch coverage path coverage	definition coverage use coverage du-path coverage

Table 2

Looking more closely at the control flow criteria, we note:

- *Node coverage* (sometimes called *statement coverage*) requires covering every node in the flowgraph.
- *Branch coverage* requires covering every edge in the flowgraph.

- *Path coverage* requires covering all paths in the flowgraph.[2]

Data flow testing is based on the principle of tracing the definition of the value of a variable through subsequent accesses to that value. Based on this idea, we say that a variable is *defined* at a node if there is a statement in that node that assigns a value to the variable (*e.g.*, through an input statement or the appearance of the variable on the left-hand side of an assignment statement). A variable is *used* if a reference is made to its contents. Such a reference can take place in a node (*i.e.*, in an output statement or on the right hand side of an assignment statement), or at a labeled edge (*i.e.*, in a predicate). Uses of the first type are called *c-uses* (for computational uses) and uses of the second type are called *p-uses* (for predicate uses).[3]

Because blocks are executed as a single unit, there is no need to worry about tracing the variable's definition and access (*definition-use pairs*) when both definition and use occur within the same block. Instead, attention is focused on *global* definitions and uses of a variable. Intuitively, a *global definition* is a definition that is used after exiting the block in which the definition occurs; a *global use* is a use of a definition from some 'earlier' block.

The remaining data flow concepts are summarized in the table below.

[2]Some of these paths may be of infinite length. Since our concern is primarily implementation oriented, we do not further consider path coverage in this paper.

[3]Note that p-uses are really found within nodes rather than edges, since the condition containing the p-use is at the tail end of the block (node) preceding multiple labeled edges. However, since we require p-uses to be properly 'covered,' (meaning that both the 'true' and the 'false' conditions are executed) most authors associate p-uses with the labeled edges rather than the preceding node, to help illustrate that both labeled edges must be executed.

Basic Data Flow Testing Concepts

- *simple path*
  A simple path is a path where every node, except possibly the first and last nodes, are distinct.
- *definition-clear path*
  Let x be a variable in program p, and let $q = (i, n_1, n_2, n_3, ..., n_k, j)$ denote a path through the flowgraph of p, starting with node i, proceeding through nodes $n_1, n_2, n_3, ..., n_k$, and terminating at node j. We say that path q is *definition-clear with respect to* x (abbreviated *def-clear*) if the (possibly empty) subpath $(n_1, n_2, ..., n_k)$ does not contain a definition of x (*i.e.*, the only definition of x, if there is one, is in the first node i).
- *global definition of a variable*
  Node i has a *global definition of* x if x is defined at i, and there is a definition-clear path from node i to some node either: (1) containing a global c-use of x; or (2) having an edge containing a p-use of x emanating from it.
- *global use of a variable*
  A *global use of* x is either a global c-use or p-use of x. A *global c-use* of x is a *c-use* of x in node i, where the value of x was not defined previously in i. (There is no need to define *global p-use*; since p-uses take place at edges, all such uses are global.)
- *du-path*
  A path $(n_1, ..., n_j, n_k)$ is a *du-path* with respect to a variable x if n_1 has a global definition of x and either: (1) n_k has a global c-use of x and $(n_1, ..., n_j, n_k)$ is a definition-clear simple path with respect to x or (2) (n_j, n_k) has a p-use of x and $(n_1, ..., n_j, n_k)$ is a definition-clear simple path with respect to x.
- *def-c-use association*
  A *def-c-use association* is a triple (i, j, x), where i is a node containing a global definition of variable x, j is a node containing a global c-use of x, and there is a definition-clear path from i to j with respect to x.
- *def-p-use association*
  A *def-p-use association* is a triple $(i, (j, k), x)$, where i is a node containing a global definition of x, (j, k) is an edge containing a p-use of x, and there is a definition-clear path from node i to edge (j, k) with respect to x.
- *def-use association*
  Either a def-c-use association or def-p-use association of x.

Table 3

A path is said to *cover a def-c-use association* (i, j, x) if it contains a definition-clear subpath from i to j with respect to x. Similarly, a path is said to *cover a def-p-use association* $(i, (j, k), x)$ if it contains a definition-clear subpath from i to (j, k) with respect to x. A path *covers a def-use association* if it covers a def-p-use association or a def-c-use association.

Formal definitions of the data flow testing criteria *definition coverage*, *use coverage*, and *du-path coverage* are presented in Table 4. Intuitively, definition coverage requires covering paths containing at least one use of every definition, use coverage requires covering paths containing every use of every definition, and du-path coverage requires covering every use of every definition *along every possible path*. Du-path coverage actually requires that every *simple* path from definition to use be executed by an adequate test set. Without the constraint of simple paths, programs with loops may have an

unbounded number of paths from a definition to a use.

Formal Definitions of the Dataflow Criteria

Given a program p and a set of test cases T, we can define definition coverage, use coverage and du-path coverage as follows:

- T satisfies *definition coverage* for p iff for every node i in p's flowgraph, and for every variable x receiving a (global) definition in i, T executes a set of paths covering *at least one* of either (1) a def-c-use association (i, j, x) or (2) a def-p-use association $(i, (j, k), x)$.
- T satisfies *use coverage* for p iff for every node i in p's flowgraph, and for every variable x receiving a (global) definition in i, T executes a set of paths covering (1) *all* def-c-use associations (i, j, x) and (2) *all* def-p-use associations $(i, (j, k), x)$.
- T satisfies *du-path* coverage for p iff, for all nodes i and j in p's flowgraph, and for every variable x receiving a (global) definition in i, T executes all du-paths from i to j such that (i, j, x) represents a def-c-use association, and all du-paths from i to (j, k) such that $(i, (j, k), x)$ represents a def-p-use association.

Table 4

Note that certain flowgraph paths may be *infeasible*, in the sense that no test data can cause those paths to be executed. We assume the definitions of [5], which only require feasible statements (branches, definitions, uses, du-paths) to be executed. Of course, the selection of test data to cover feasible components is undecidable [19], making the application of these criteria undecidable.

2.3 Flowgraph-Based Class Testing

Our discussion of flowgraph-based class testing in this section follows from the work of [20], whose flowgraph model can be stated as follows. A node in the graph represents an operation; an edge between operation A and operation B means that it is permissible for a client module to invoke operation A followed by operation B. Determination of whether or not an edge exists is based on the model-based specification for the class, *i.e.*, on whether the *requires* clause of B can be satisfied given the *ensures* clause of A. In the previous example, there would be no edge from `NEW` to `POP`, since `NEW` returns an empty stack, violating the requires clause of `POP`.[4] Since the conjunction of arbitrary predicate calculus expressions is undecidable, membership in the set of edges in a class flowgraph is undecidable. This is much like the undecidability problem for restricting conventional flowgraph coverage criteria to feasible code.

Certain operations are *control* operations, *i.e.*, they return `true` or `false` and can be used to construct conditions in client modules [20]. For such operations, there are two labeled edges (a '`true`' edge and a '`false`' edge) emanating from them to every operation that can follow them in a

[4] Note that the requires clause and the ensures clause for two different operations are written in terms of variables that are local to those operations. However, [20] assumes that if the parameter lists of the two operations contain common types, the same actual parameters are used for both operations. Evaluation of the ensures-requires conjunction takes advantage of this observation. For example, in establishing that there is no edge from `NEW` to `POP`, the *same* stack is used in the ensures clause of `NEW` as in the requires clause of `POP`.

client program. For example, for the control operation `IS_EMPTY` found in some stack classes, there is a '`true`' edge emanating from `IS_EMPTY` to every operation that a client is permitted[5] to invoke after `IS_EMPTY` returns true, and a 'false' edge emanating from `IS_EMPTY` to every operation that a client is permitted to invoke after `IS_EMPTY` returns false. Control operations thus have the same role in the class flowgraph as conditions have in a conventional program flowgraph.

The testing criteria defined for conventional program flowgraphs can now be viewed as defining sequences of operations based on the class flowgraph. The class analog for node coverage requires that a sequence or sequences containing every operation be executed, and the class analog for branch coverage requires that a sequence or sequences containing every edge in the class flowgraph be executed. Class analogs for the data flow criteria can be provided similarly, assuming that the underlying data flow concepts (*i.e.*, definitions, uses, def-use associations, *etc.*) are well-defined for classes. To begin with, we note that all definitions and uses are associated with class operations, rather than statement blocks. A '*p-use*' is a use associated with a control operation, rather than a condition. Since there are multiple labeled edges emanating from control operations (just as there were from blocks containing conditions), all of the concepts regarding p-uses are the same as before. In particular, each labeled edge emanating from a control operation represents a p-use to be exercised. A '*c-use*' is a use within any other (non-control) operation.

The other general observation that we make about definitions and uses in this context is that data flow concepts must be interpreted in the context of *types* rather than strictly in terms of *variables* (as is the case for conventional programs). Operations define and use their parameters, *e.g.*, a `PUSH` operation defines and uses *any* actual stack instance with which it is parameterized in a client module. Thus, in this example, data flow analysis takes place with respect to type "stack" in general, rather than with respect to a specific variable. Therefore, def-use associations for classes are triples involving (a) the node in which a type is defined, (b) the node or edge where a type is used and (c) the type itself.

The rules introduced in [20] for locating definitions and uses can be summarized as follows. An operation is said to contain a *definition of a type* if a formal parameter to the operation appears of that type, and the ensures clause specifies that the parameter is to be changed. Specifically, if the symbol x appears (without '#') in the ensures clause, and the predicate $x = \#x$ is absent [6], then [20] assumes that x is defined by the operation (the absence of # means that there is a 'new' value for x). An operation is said to contain *a use of a type*[7] if a formal parameter to the operation appears of that type, and the ensures clause specifies that the parameter is used somehow to produce a result. Specifically, if the symbol $\#x$ appears in the ensures clause in any predicate other than $x = \#x$ (which is considered neither a definition or a use of x), then x is considered to have been used (since its 'old' value is a part of some predicate that represents the computation of the operation).

[5] based on the ensures-requires conjunction in the specification

[6] The predicate $x = \#x$ in an ensures clause implies that the formal parameter x is not changed by the operation; the absence of this predicate implies that x could be changed.

[7] Or, in the case of a p-use, the edges emanating from the operation are said to contain p-uses of the type.

A very simple example will serve to clarify these ideas. Consider the model-based specification of a stack module presented earlier. This module has four operations: `NEW`, `PUSH`, `POP` and `TOP`. Because of domain restrictions on `POP` and `TOP`, there is no edge drawn from `NEW` to either of these operations. Otherwise, every pair of operations is connected by a single edge (none of the operations are control operations requiring multiple edges), and there is a reflexive edge from every operation to itself. This is illustrated in Figure 1.

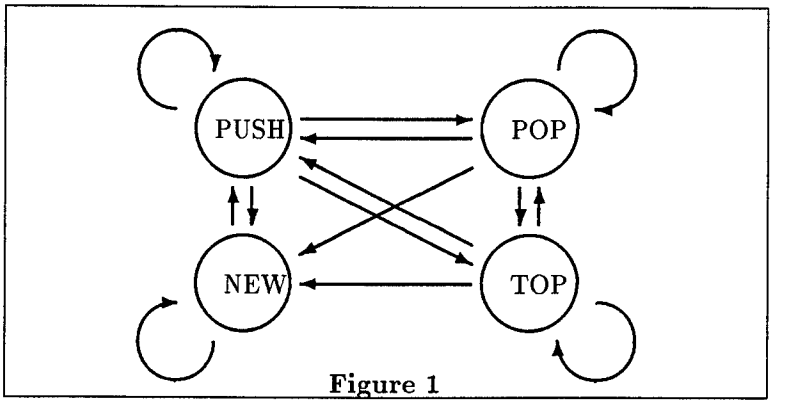


Figure 1

The following sequences represent one way of satisfying node coverage and branch coverage:

Criterion	Sequence
node	`NEW; PUSH; TOP; POP`
branch	`NEW; NEW; PUSH; NEW; PUSH; PUSH; POP; PUSH; TOP; TOP; PUSH; POP; POP; NEW; PUSH; PUSH; PUSH; POP; TOP; POP; TOP; NEW`

To define sequences satisfying the data flow coverage criteria, we must consider the definitions and uses found in each operation. Using the rules above, we have the following set of definitions and uses:

Operations	Defines	Uses
NEW	stack	
PUSH	stack	stack, T
POP	stack	stack
TOP	T	stack

Then the following sequences represent one way of satisfying definition, use and du-path coverage:

Criterion	Sequence
definition	`NEW; PUSH; PUSH; POP; TOP; PUSH`
use	`NEW; PUSH; PUSH; POP; POP; PUSH; TOP; PUSH; POP TOP`
du-path	`NEW; PUSH; PUSH; POP; POP; PUSH; TOP; PUSH; TOP POP; TOP; POP; PUSH; TOP; PUSH; TOP; NEW; PUSH PUSH; TOP; POP; PUSH`

For a more complex example involving control operations, the reader is referred to [20].

3 Foundational Theory

In this section, we consider the theoretical contribution of this work to ADT sequence testing. We first consider a method for modeling ADTs using flowgraphs, providing a well-understood foundation for ADT testing criteria based on flowgraphs (much like the foundation for flowgraph-based testing of conventional programs). We then provide a series of theorems that contain algorithms for generating operation sequences based on these techniques.

3.1 A General Class Graph Model

In this section, we develop a formal framework for modeling classes with flowgraphs. A significant side-effect of this framework is that model-based specifications are unnecessary to utilize flowgraph-based sequence testing techniques, such as those proposed by [20]. In fact, we will argue that formal specifications are not necessary *at all* to utilize these techniques. Additionally, the framework provides a basis for deriving results concerning automation of the techniques.

We have the following basic definition:

> A *class graph* is a collection $< N, E, D, U, I >$, where N refers to the set of all nodes, E refers to the set of all edges, D refers to the set of all definitions, U refers to the set of all uses, and I refers to the set of all infeasible subpaths. Δ

A class graph is a formalization of the information that is needed to apply flowgraph-based testing techniques to classes. As a starting point, consider an interpretation of the definitions from [20] in the context of this framework:

- N (the set of operations) is determined by looking at the set of operations present in the (model-based) specification.
- E consists entirely of those edges that are 'feasible' according to the specification (*i.e.*, those edges from operation A to operation B, where the ensures clause of A is consistent with the requires clause of B). To provide a better foundation for our later observations, however, we eliminate feasibility from the requirements for E, and assume that E contains at least one edge between every pair of operations, and two edges (labeled `true` and `false`) from every control operation to every other operation. Thus, the derivation of E is entirely mechanical, and does not involve processing semantic information from the specification.
- D and U are determined by following the rules outlined in the previous section to identify definitions and uses from the specification. We assume that definitions and uses each consist of pairs (*operation*, *type*), where *operation* is the operation in which the definition or use is occurring, and *type* is the type of the parameter involved in the definition or use.
- I is composed of subpaths that are infeasible according to the specification.

With this modification to E, we note that I contains both infeasible edges as well as longer infeasible subpaths. For example, the edge from NEW to POP is in I, given the earlier stack specification. However, the subpath <NEW; PUSH; PUSH; POP; POP; POP> is also infeasible for the same reason that <NEW; POP> is infeasible. Both <NEW; POP> and <NEW; PUSH; PUSH; POP; POP; POP> are in I, given the earlier stack specification.

In defining a general theory for class graphs, it is important to distinguish between four distinct forms of infeasibility that together constitute the set I. We refer to these as *Type 1*, *Type 2*, *Type 3*, and *Type 4* infeasibility, shown in Table 5.

Types of Infeasibility for Class Graphs

- **Type 1:** Subpaths are *type 1 infeasible* if according to the specification, the subpath *should not* be executed. This is the form of infeasibility identified above. The sequences <NEW; POP> and <NEW; PUSH; PUSH; POP; POP; POP> are type 1 infeasible, based on the specifications in Table 1.
- **Type 2:** Subpaths are *type 2 infeasible* if according to the specification, the subpath *cannot* be executed. This type of infeasibility only arises for subpaths containing control operations with labeled edges. For example, given a standard stack specification, the subpath <NEW; PUSH; PUSH; POP; IS_EMPTY>, followed by an edge labeled 'true' (and leading to any other operation) is type 2 infeasible. This type of infeasibility can be established by sequentially applying the ensures clauses one after the other, and noting that the value of the ensures clause of IS_EMPTY is inconsistent with the value of the 'true' labeled edge that follows.
- **Type 3:** Subpaths are *type 3 infeasible* if *when attempting to execute the subpath*, the subpath cannot be executed because the value of a control operation in the subpath is inconsistent with the intended labeled edge that follows. This is the same problem as type 2 infeasibility, except this type of infeasibility is a result of the *implementation* and not the specification.
- **Type 4:** Subpaths where there is a crash in the middle of the subpath are *type 4 infeasible*. For example, consider the sequence <NEW; PUSH; POP; POP; POP>. Suppose that POP crashes on an empty stack. Then this sequence cannot be executed, because there will be a crash after the second invocation of POP.

Table 5

Note that types 1 and 2 infeasibility are properties of the *specification*, while types 3 and 4 infeasibility are properties of the *implementation*. Since I can contain subpaths associated with all four of these types of infeasibility, we refer to subsets of I corresponding to each type as I_1, I_2, I_3 and I_4, respectively. Note also that membership in each of these subsets is undecidable. Also, when the implementation is correct, there are certain relationships between these subsets, namely:

- $I_2 = I_3$ Subpaths in both of these sets are related in that the next-to-last operation is a control operation, followed by a labeled edge. The labeled edge that is executed in the implementation(and conversely, the labeled edge that is not executed, and therefore infeasible) should be the same edge that the specification implies *should be* (or *should not be*) executed.

- $I_4 \subseteq I_1$ Any subpath for which the implementation crashes should be a path that the specification implies *should not* be executed by a client. Otherwise, the implementation is incorrect, in that it crashes on a legitimate sequence of operations. However, the fact that a subpath should not be executed does not imply that a correct implementation *must crash* on such a subpath.

Of course, when the implementation is incorrect, one cannot guarantee any particular relationship between the specification and implementation forms of infeasibility (without knowing anything about why the implementation is incorrect).

One important design decision in developing classes relates to whether modules are designed to be *defensive* or *nondefensive*. In the above framework, a defensive module is a module *without type 1 infeasible subpaths*. In other words, the specification is written in such a way that there are no domain restrictions placed on any operations, *i.e.*, the module is designed to be 'error-catching.' An example of a defensive design is to require a particular defined behavior for the `POP` operation on an empty stack input (*i.e.*, either a no-op or an exception). A nondefensive module is just the opposite, *i.e.*, a module where domain restrictions are placed on any or all operations.

Our discussion so far has been centered around the possibility that modules might be nondefensive. Much of the literature has suggested that modules should always be defensive, thus eliminating any possibility of Type 1 infeasibility [3, 7, 9, 4]. This is not the universal view, however, in that at least [18] and [10] both imply that there are at least some cases where nondefensive modules are desirable. The resolution of this issue is outside the scope of this paper, but will at least maintain the distinction between defensive and nondefensive modules for the remainder of the discussion.

We are now in a position to re-examine the question raised at the beginning of this section concerning the applicability of the flowgraph-based criteria to classes without model-based specifications. To apply the flowgraph-based criteria, a class graph must be constructed. In constructing such a graph, we first observe that although N, E, D and U are determined from a model-based specification in [20], this is not strictly necessary in most cases. For most object-oriented languages, N, E, D and U are obtainable from the class interface (*i.e.*, from the implementation), provided that the class interface has the following properties:

- All operation names are listed.
- For each operation, all parameters must be listed. For each parameter, both *type* and *transmission mode* must be listed. (Possible transmission modes are IN, OUT and IN-OUT, each with the obvious meaning.)

In most object-oriented languages, the class interface satisfies these properties. Given these properties, the construction of both N and E is trivial, simply by using the names of the operations, and by interpreting operations that return booleans as control operations (and therefore constructing labeled edges to emanate from those operations). D and U are constructed as follows. For every IN or IN-OUT parameter to any operation, a definition (*i.e.*, a pair consisting of *(operation, type)*)

is added to set D. Similarly, for every OUT or IN-OUT parameter to any operation, a use (*i.e.*, a pair consisting of *(operation, type)*) is added to set U.

Using this technique, N, E, D and U may be constructed from the implementation alone. In addition, even if the specification is used to derive N, E, D, and U, we note that it may not be necessary for the specification to be model-based. Most styles of algebraic specifications provide this information as well, provided the specification has a "syntax" section denoting the signatures of the operations (as was present in the example from Table 1). Thus, the bottom line is that N, E, D and U can be determined with or without the existence of formal specifications, and regardless of whether the specification is model-based.

To complete the construction of the class graph, we must also be able to construct I as well. Our objective in applying the desired testing techniques is to test the class over sequences that satisfy certain criteria (*i.e.*, coverage of all feasible branches, definitions, uses, *etc.*). The purpose in constructing I is to allow us to eliminate these sequences that are inappropriate or impossible to test. By their nature, I_3 and I_4 are based on properties of the implementation, and so specifications are not used in deriving these sets. Moreover, such sequences can never be executed (by definition), and so there is no need to consider their explicit elimination.

This only leaves I_1 and I_2. We first observe that eliminating the subpaths in I_2 is not a desirable thing to do. Assuming a correct implementation, subpaths in I_2 are impossible to test anyway (since they are also in I_3). However, if there is a subpath that is in I_2 and not in I_3, then by our earlier observation that I_2 should equal I_3, the class is incorrect with respect to that subpath. Testing over such a subpath is a desirable thing to do, and we don't want to eliminate such subpaths from our testing process.

Finally, we consider I_1. I_1 is defined by domain restrictions on the class operations, presumably from within the specification. Sequences in I_1 represent unspecified behavior, which we assume is unnecessary to test [12, 13, 20]. We would therefore like to identify such sequences and eliminate them from testing. We note that provided domain restrictions are present in some discernible form within the specification, it is certainly unnecessary to require the specification to be model-based in order to capture such restrictions. For example, the algebraic specification example in Section 2 contained explicit domain restrictions. Further, we assume that in the absence of domain restrictions (regardless of specification paradigm), the module design is intended to be defensive ***by default***. Without a way to communicate to prospective clients that certain domain restrictions are in place, a client is unable to knowledgeably utilize a nondefensive module. The bottom line is that in testing, we must utilize the specification (if one exists), to find I_1, although there is no reason that the specification has to be model-based. However, in the absence of a specification, the question of finding I_1 is eliminated, because the module is (by default) defensive, and I_1 is empty.

In summary, to find the class graph (and apply the graph-based testing criteria), the specification can be ignored when finding N, E, D, and U if desired. The specification, if it exists, must be used in finding I (specifically, in finding I_1). However, if no specification exists, as may be the case given

current development practice, then the specification is no longer needed in finding I. Consequently, it is still possible to find the class graph without the specification and apply flowgraph-based testing techniques.

3.2 Implementation Theorems

Here we consider the extent to which sequences of operations satisfying various graph-based criteria may be automatically generated. There are two major barriers to automation. One problem involves infeasible paths within the class graph, since finding such paths is undecidable. Consequently, it is impossible to avoid automatically generating sequences that contain such paths. A related problem is that even without infeasible paths, it is impossible to generate a sequence containing subpaths that drive the different conditions returned by control operations.

To deal with these problems, we wish to consider ways of eliminating infeasible paths from the class graph (*i.e.*, from I). This will be accomplished in two ways. First, we restrict our attention to class modules *that are designed defensively*. That is, there are no type 1 infeasible subpaths in the class graph (*i.e.*, I_1 is empty). Note that we are not saying that all class modules *should be* designed defensively, but we are assuming that there are situations where modules *are* designed defensively. We are later able to show how to relax this assumption and allow modules to be designed nondefensively, although we are not able to completely automate the sequence generation process under such circumstances (given the undecidability problems).

The second way that we eliminate infeasible paths is to perform a transformation on class graphs as follows: *Replace every pair of labeled edges with a single unlabeled edge*. This automatically changes the testing criteria in the obvious way: a criterion that required coverage of either or both of a true/false pair of labeled edges now only requires coverage of the corresponding single unlabeled edge. We call this new version of an class graph a *weak class graph*, and the corresponding versions of the testing criteria *weak criteria* (*i.e.*, *weak node coverage*, *weak branch coverage*, *weak definition coverage*, *weak use coverage*, and *weak du-path coverage*). We note that weak node coverage is equivalent to (strong) node coverage, since node coverage does not require covering particular edges; however, the other criteria are strictly weaker than their corresponding strong counterparts.

In defining weak class graphs along with the corresponding testing criteria, we have effectively eliminated the infeasibility barriers to automatic generation of sequences to satisfy the criteria. As discussed above, I_1 is empty because of our restriction to defensive modules. For weak class graphs, it should be evident that both I_2 and I_3 are empty, in that these types of infeasibility are associated with labeled edges, which no longer exist. We note that I_4 may be non-empty, but for defensive modules, any subpaths in I_4 are defect-revealing (since such subpaths represent crashes), and their inclusion is therefore desirable. We have also eliminated the problem that it is impossible to generate subpaths to drive all of the conditions associated with a control operations, since it is now only necessary to cover one such condition.

In our results, we will be interested in showing not only that it is possible to efficiently gener-

ate operation sequences satisfying the criteria, but that it is possible to generate *minimum length* operation sequences satisfying the criteria. To this end, it is also necessary to place two additional restrictions:

- We exclude from consideration class modules containing operations involving multiple definitions and multiple uses of the class (*e.g.*, a class containing an operation $Equal(s1, s2)$, where $s1$ and $s2$ contain instances of the type).
- In applying the dataflow-based criteria, we restrict our attention to dataflow related to the type exported by the class being tested. That is, when constructing D and U in the weak class graph, an operation is assumed to contain a definition (use) only if it defines (uses) an instance of the class being tested. Definitions and uses involving other types are ignored. Note that we do not exclude classes containing operations involving 'foreign' types; rather, we simply ignore such types in the dataflow analysis.

Under these conditions, we are now able to show that it is possible to efficiently generate minimum length sequences of operations that satisfy most of the weak criteria (except weak du-path coverage) for defensive modules. We also are able to state, in each case, the exact length of such a minimum length sequence as a function of the number of operations. Before discussing these results, we note that our use of these weak versions of the criteria, as well as the two additional assumptions above, substantially weaken the degree of testing demanded. This would appear to reduce the significance of our results. However, testing to satisfy the weak criteria at least tests certain interesting combinations of operations, and has the virtue of being fully automatable. Moreover, in all cases, it is easy to see that a sequence that satisfies a weak criterion is simply a subsequence of the sequence required to satisfy its strong counterpart. Thus, satisfaction of the weak criteria can be viewed as a starting point for achieving the strong criteria, providing the ability to at least automate a subset of the test generation process. Since one of our goals was that of automation, our work on the prototype tool (described in Section 4) is based on the weak versions of the criteria.

Because of our restrictions above involving dataflow between operations, we are able to classify operations into three categories: *pure OUT* operations that define a class instance without using it, *pure IN* operations that use a class instance without defining it, and *IN OUT* operations that both use and define the class instance. In the theorems below, we assume there are n total operations in the class, of which there are a pure OUT operations A_1, A_2, ..., A_a, b IN OUT operations B_1, B_2, ..., B_b, and c pure IN operations C_1, C_2, ..., C_c. We assume that $a \geq 1$, since there must always be at least one pure OUT operation at the beginning of a sequence to ensure that the class instance is defined before it is used. We also assume that there is at least one operation that uses the class, *i.e.*, that $b + c \geq 1$. We would expect any legitimate class to have these satisfy these restrictions on a, b and c, so these assumptions are made without loss of generality.

We refer to a pure OUT or an IN OUT operation as a *definition operation*, and a *use operation* denotes either an IN OUT or a pure OUT operation. Also, we assume that there is only one instance

of one type of class under consideration. Our results are summarized as follows.

Theorem 4.1. A minimum length sequence of operations that satisfies weak node coverage can be found in polynomial time.

Proof. The following sequence contains each operation exactly one time, for a total of $n = a+b+c$ operations.

$$A_1, A_2, \ldots, A_a, B_1, B_2, \ldots, B_b, C_1, C_2, \ldots, C_c.$$

Obviously, this is a minimum length sequence that satisfies weak node coverage. It can be constructed in $\mathcal{O}(n)$ time. □

Theorem 4.2. A minimum length sequence of operations that satisfies weak branch coverage can be found in polynomial time.

Proof. The flow graph has $n = a + b + c$ nodes, each of which has indegree n and outdegree n, so the flow graph is Eulerian. There are a total of n^2 arcs, so the minimum length sequence must have at least $n^2 + 1$ operations.
Denote the n nodes by $V_1, V_2, \ldots, V_n$, where V_1 is a pure OUT operation that is the desired starting and ending vertex for an Eulerian cycle. The following sequence forms an Eulerian cycle with exactly n^2 arcs and $n^2 + 1$ operations.

$$\begin{array}{l} V_1, V_1, \\ V_2, V_2, V_1, \\ V_3, V_3, V_2, V_3, V_1, \\ V_4, V_4, V_3, V_4, V_2, V_4, V_1, \\ V_5, V_5, V_4, V_5, V_3, V_5, V_2, V_5, V_1, \\ \qquad \ldots \\ V_n, V_n, V_{n-1}, V_n, V_{n-2}, V_n, \ldots, V_n, V_3, V_n, V_2, V_n, V_1. \end{array}$$

This sequence satisfies weak branch coverage, and can be constructed in $\mathcal{O}(n^2)$ time. □

Theorem 4.3. A minimum length sequence of operations that satisfies weak definition coverage can be found in polynomial time.

Proof. Each occurrence of a use operation corresponds only to the most recently preceding definition operation, so there are at least $a + b$ use operations in the sequence. Also, each pure OUT operation appears at least once, which requires a additional operations in the sequence. So the minimum length sequence must have at least $2a + b$ operations.
The following sequence satisfies weak definition coverage and has exactly $2a+b$ operations.

$$A_1, C_1, A_2, C_1, \ldots, A_{a-1}, C_1, A_a, B_1, B_2, \ldots, B_b, C_1.$$

[As a special case, if $c = 0$, then substitute B_1 for all occurrences of C_1.] This sequence can be constructed in $\mathcal{O}(n)$ time. □

Theorem 4.4. A minimum length sequence of operations that satisfies weak use coverage can be found in polynomial time.

Proof. There are $a + b$ definition operations, each of which has $b + c$ uses. Each occurrence of a use operation corresponds only to the most recently preceding definition operation, so there are at least $(a + b)(b + c)$ use operations in the sequence. Also, each pure OUT operation appears at least b times (once leading to each IN OUT operation), which requires ab additional operations in the sequence. So the minimum length sequence must have at least $ab + (a + b)(b + c)$ operations.
First consider the following sequence s with $2ab$ operations.

$$\begin{array}{llll} A_1, B_1, & A_1, B_2, & \ldots, & A_1, B_b, \\ A_2, B_1, & A_2, B_2, & \ldots, & A_2, B_b, \\ & \ldots & & \\ A_{a-1}, B_1, & A_{a-1}, B_2, & \ldots, & A_{a-1}, B_b, \\ A_a, B_1, & A_a, B_2, & \ldots, & A_a, B_b. \end{array}$$

Let d denote the sequence $C_1, C_2, \ldots, C_c$ with c operations. Let e denote the sequence with $b^2 + 1$ operations formed by an Eulerian cycle through $B_1, B_2, \ldots, B_b$, starting and ending at B_b. [e can be found as described in Theorem 4.2.] Next insert d into s after the first occurrence of each A_i and each B_j, and substitute e for the final B_b in s. This yields the sequence

$$\begin{array}{llll} A_1, d, B_1, d, & A_1, B_2, d, & \ldots, & A_1, B_b, d, \\ A_2, d, B_1, & A_2, B_2, & \ldots, & A_2, B_b, \\ & \ldots & & \\ A_{a-1}, d, B_1, & A_{a-1}, B_2, & \ldots, & A_{a-1}, B_b, \\ A_a, d, B_1, & A_a, B_2, & \ldots, & A_a, e. \end{array}$$

This modified sequence satisfies weak use coverage and has exactly $2ab + b^2 + ac + bc$ operations. This sequence can be constructed in $\mathcal{O}(n^2)$ time. □

Theorem 4.5. A minimum length sequence of operations that satisfies weak du-path coverage can neither be found nor tested in polynomial time.

Proof. Consider the special case in which $a = 1$ and $b = 0$, so that the number of distinct operations is given by $n = 1 + c$. But the number of distinct simple paths of length c leading from definition A_1 is $c! = (n-1)!$, and each operation in any given sequence is the final operation of at most one path of length c. Therefore no polynomial length sequence that satisfies weak du-path coverage can exist. □

We conclude this section by noting some results concerning the relationships among the criteria. The *subsumption* relation is commonly used to compare testing criteria [5, 6], and can be defined for the sequence testing criteria as follows (as in [20]): a sequence testing criterion $C1$ ***subsumes*** $C2$ if, for every class module and specification, every sequence satisfying $C1$ also satisfies $C2$. The following theorem is then trivial:

Theorem 4.6. The following subsumption relationships hold:
(a) Node coverage subsumes weak node coverage.
(b) Branch coverage subsumes weak branch coverage.
(c) Definition coverage subsumes weak definition coverage.
(d) Use coverage subsumes weak use coverage.
(e) Du-path coverage subsumes weak du-path coverage. □

Perhaps the more interesting observation concerns the subsumption relationships among the different weak criteria. With the strong criteria, use coverage and branch coverage are incomparable,

while use coverage subsumes both definition and node coverage [20]. It is also shown in [20] that du-path coverage is incomparable to all of the other criteria.[8] These relationships are depicted in Figure 2 below. [9]

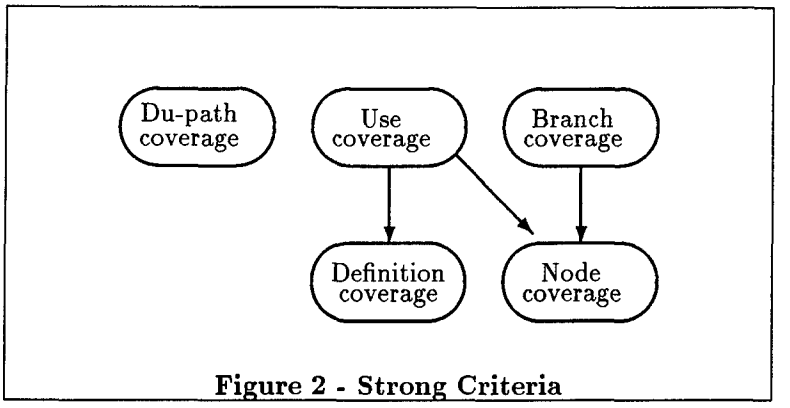


Figure 2 - Strong Criteria

With the weak criteria, there are some additional subsumptions where none held before. In particular, du-path coverage subsumes use coverage because there is always a feasible edge from every definition to every use, so there is always a feasible simple path from every definition to every use. Hence, it is necessary to cover every definition-use association with du-path coverage. In addition, weak branch coverage subsumes weak use coverage, since every definition-use association may be covered on a single branch (*i.e.*, there is a branch between every definition node and every use node, and it is not necessary to follow that branch with specific labeled edges in order to cover the definition-use association). The other subsumptions and non-subsumptions hold correspondingly for the strong and weak analogs. The relationships between the weak criteria are depicted in Figure 3 below.

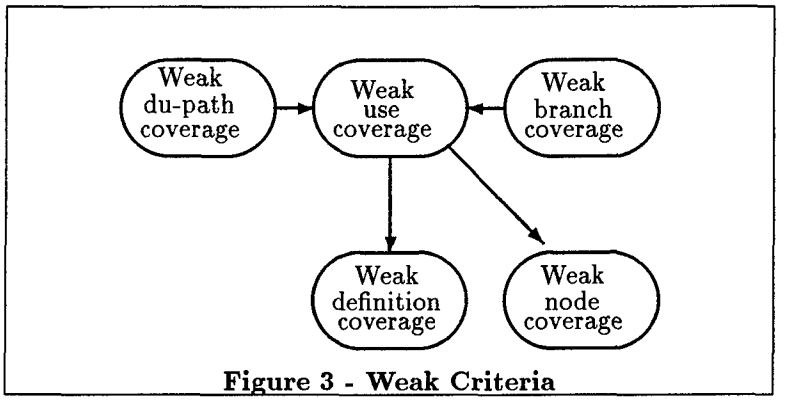


Figure 3 - Weak Criteria

[8] If the only path between a definition and a use contains a cycle, then du-path coverage does not require coverage of that definition-use association, which may leave certain branches and nodes uncovered as well. However, clearly the other criteria do not subsume du-path coverage either, since in general, du-path coverage is substantially more demanding.

[9] Note that these relationships differ from the corresponding relationships among the analogous conventional testing criteria developed in [5]. The details involving this discrepancy are discussed in [20].

Note that because subsumption is transitive, weak branch coverage and weak du-path coverage both subsume weak node coverage, even though there are no explicit edges associated with these relationships in Figure 3. Also, note that all of the weak criteria (except weak definition coverage) subsume *strong* node coverage. As pointed out earlier, weak node coverage and strong node coverage are equivalent (*i.e.*, each subsumes the other).

4 Prototype Implementation

In this section, we discuss the design of the driver generator tool (called `dtest`), as well as the basic framework within which test execution occurs.

4.1 Tool Design

`Dtest` is currently functional for classes implemented within Ada packages. As part of its sequence testing methods, it includes the weak forms of branch coverage and use coverage. As such, formal specifications are not required, and test generation is completely automatic. To compensate for some of the deficiencies of the weaker forms of the criteria, `dtest` provides the tester with the capability of reviewing previous tests and manually augmenting the test sequence to cover additional combinations of operations not covered by the automatically generated sequence.

`Dtest` is invoked from the MS-Windows environment. There are three major windows involved in driver generation:

- *The file manager window.* In this window, the user is permitted to select the Ada package to be tested. The user has access to the DOS directory hierarchy through the standard directory navigation tool found in many DOS/Windows applications. Once a file containing an Ada package is selected, the user may view the contents of the file, view a log containing information regarding previous tests that have been run on the package, or proceed to the next major window. The tester can also retrieve, view, compile, or execute an existing driver through this window.

- *Initialization/display operation selection window.* In this window, the user is permitted to select initialization and display operations for the class under test. The initialization operation is invoked at the beginning of driver execution; the display operation is invoked after every other operation in the sequence to check the correctness of the state of the object. A list of potential initialization/display operations is provided for the user to choose from. Once these operations have been selected, the user clicks "continue" to proceed to the next window.

- *Testing mode selection window.* In this window, the user is permitted to select the testing method. There are three choices currently implemented:

- (Weak) Edge coverage
- (Weak) Use coverage
- Interpreter mode

As discussed above, the two coverage criteria are the weak forms of the criteria, and choosing those options results in generating a driver that executes the sequences of operations mandated by the respective criteria. Interpreter mode permits the tester to select individual operations to be tested (*i.e.*, one at a time). Interpreter mode allows classical coverage strategies to be applied to individual operations (as opposed to using coverage strategies to select sequences of operations, as is the case with the first two options).

Along with selecting a testing method, the user may use this final window to select a subset of operations to test. The default testing is to utilize all of the operations that are currently available in the class when satisfying the specified coverage criterion. However, the third window may be used to select a subset of operations to be included in any test, allowing the user to focus strictly on those operations. This is particularly useful when an error shows up as a result of testing a larger sequence, and the tester wishes to examine specific operations in this sequence that appear to be problematic.

4.2 Test Execution

Once a driver is generated, it must be compiled with the package containing the class under test. Once compiled, it must be executed, and the resulting execution behavior must be observed. The tester can return to the first window to compile and execute a particular driver.

Executing a class operation involves three steps:

1. Obtaining initial values for all IN parameters to the operation.
2. Executing the operation.
3. Displaying the final values of all OUT parameters to the operation.

In obtaining initial values for IN parameters, there are two categories of such parameters to consider. We classify all parameters as either *native* or *foreign*: a native parameter is a parameter whose type is that of the class being exported, while a foreign parameter is any other parameter used in the operation. [10] For native parameters, we assume that a single class instance is propagated through a sequence of operations. That is, for a sequence of operations `<A;B>`, the object produced as an OUT parameter by `A` is used as the IN parameter for `B`. In this way, we can study the cumulative effect as a sequence of operations modifies an instance of the class under test. We assume that all

[10] In the operation `ENQUEUE(q: in out QUEUE; e: in ITEM)`, `q` is native, while `e` is foreign.

foreign parameters are built-in (*e.g.*, integers), and so there is a built-in input operations to obtain the values of such parameters.

After the operation is executed, the final values of all OUT parameters must be displayed in order to permit the tester to evaluate the correctness of the operation. For native parameters, the display operation specified by the user is used to display the object; for (built-in) foreign parameters, the appropriate built-in `PUT` operation is used. Display operations are invoked for OUT parameters after every operation that is tested. The driver alternates between tested operations and display operations when executing the class.

As a complete example, suppose a sequence testing criterion mandates execution of the sequence `<A; B; A>`. The driver will have to (a) obtain values for foreign IN parameters before each operation, and (b) display the OUT parameters after each operation. Thus, the driver will actually execute this sequence as follows:

```
-----------------------------------------
Obtain values for A's foreign IN parameters
Execute A
Display A's OUT parameters
-----------------------------------------
Obtain values for B's foreign IN parameters
Execute B
Display B's OUT parameters
-----------------------------------------
Obtain values for A's foreign IN parameters
Execute A
Display A's OUT parameters
-----------------------------------------
```

Below we provide examples of executing both types of drivers generated by `dtest`: (a) a driver satisfying a particular coverage criterion, and (b) a driver functioning as an operation interpreter. We first consider a driver in category (a), that is executing the sequence `<ENQUEUE; ENQUEUE; DEQUEUE>` from a queue class. (Assume that this is part of a larger sequence needed to satisfy some coverage criterion.) Note that the display operation adopts the convention of displaying the queue as a string of values separated by commas, with the head of the queue at the left end of the string. To provide a more interesting example, we assume that `DEQUEUE` incorrectly removes an element off the *back* of the queue, rather than off the front. We have the following (user inputs are shown in italics).

```
--- TESTING ENQUEUE(q:  in out QUEUE; e:  in INTEGER) ---
Given initial value for q: []
Enter initial value for e: 3
Result of ENQUEUE:
    q = [3]
Hit return to continue...

--- TESTING ENQUEUE(q:  in out QUEUE; e:  in INTEGER) ---
Given initial value for q: [3]
Enter initial value for e: 2
Result of ENQUEUE:
    q = [3,2]
Hit return to continue...

--- TESTING DEQUEUE(q:  in out QUEUE; e:  out INTEGER) ---
Given initial value for q: [3,2]
Result of DEQUEUE:
    q = [3]
Hit return to continue...
```

Second, we consider a driver functioning as an operation interpreter. Assume that the operations available are **ENQUEUE**, **DEQUEUE**, and **IS_EMPTY**. Then we have the following:

```
Select an operation to test:
(1) ENQUEUE(q:  in out QUEUE; e:  in INTEGER);
(2) DEQUEUE(q:  in out QUEUE; e:  out INTEGER);
(3) IS_EMPTY(q:  in QUEUE) returns BOOLEAN;
Enter a selection? 1

--- TESTING ENQUEUE(q:  in out QUEUE; e:  in INTEGER) ---
Given initial value for q: []
Enter initial value for e: 3
Result of ENQUEUE:
    q = [3]
Hit return to continue...

Select an operation to test:
(1) ENQUEUE(q:  in out QUEUE; e:  in INTEGER);
(2) DEQUEUE(q:  in out QUEUE; e:  out INTEGER);
(3) IS_EMPTY(q:  in QUEUE) returns BOOLEAN;
Enter a selection? 2

--- TESTING DEQUEUE(q:  in out QUEUE; e:  out INTEGER) ---
Given initial value for q: [3]
Result of DEQUEUE:
    q = []
Hit return to continue...
```

Future planned enhancements to **dtest** include other methods for checking test results, including the use of formal specification-based methods, as well as test oracle files that help drive the types of sequences generated and permit results to be checked automatically. However, even with an automated oracle, display operations are essential during the debugging process. In the above

example, by displaying the value of the queue after each operation, the problem with **DEQUEUE** is clearly outlined. With automated output checking, it would be possible to determine that an error occurred, but the source of the error might be unknown.

5 Conclusion

In this project, we have developed a tool that could be used as part of an overall strategy to support the certification of reusable software components. The tool provides support for testing object-oriented class modules, implemented in Ada packages. Our tool supports individual testing of class operations (through the interpreter mode) as well as testing sequences of class operations (based on systematic, flowgraph-based criteria). We have provided part of a foundational theory for future work in the area of flowgraph-based class (sequence) testing criteria, by providing a set of fundamental terms and concepts in the area (much like the fundamental terms and concepts in flowgraph-based testing for conventional programs).

Our tool provides a framework for testing object-oriented class modules in general, that can be extended beyond just the work here. In particular, we expect this same interface to be useful to a much larger variety of testing strategies than was reported here. Future work will involve extending the tool to support new testing strategies, as well as conducting tests on the effectiveness of the strategies proposed here.

References

[1] "ASSET Library Repository Catalog," SAIC Corp., April 1992.

[2] Begley, A., et al. "Ada Test and Verification System (ATVS): Software Requirements Specification," General Research Corporation, 1988.

[3] Berard, E.V. "Creating Reusable Ada Software," EVB Software Engineering, Frederick, Maryland, 1987.

[4] Booch, G. *Software Components with Ada*, Benjamin/Cummings, Menlo Park, California, 1987.

[5] Frankl, P. and E. Weyuker. "An Applicable Family of Data Flow Testing Criteria," *IEEE Trans. on Software Engineering,* SE-14, 10, October 1988, pp. 1483-1498.

[6] Gourlay, J. "A Mathematical Framework for the Investigation of Testing," *IEEE Trans. on Software Engineering,* SE-9, 6, November 1983, pp. 686-709.

[7] Hibbard, P, Hisgen, A., Rosenberg, J. and Sherman, M. "Programming in Ada: Examples" in *Studies in Ada Style*, pp. 35-101, Springer-Verlag, New York, 1983.

[8] Hooper, J. and R. Chester. "Software Reuse Guidelines," Technical Report ASQB-GI-90-015, U.S. Army Institute for Research in Management Information, Communications and Computer Sciences (AIRMICS).

[9] Liskov, B. and Guttag, J. *Abstraction and Specification in Program Development*, McGraw-Hill, New York, 1986.

[10] Meyer, B. *Object-Oriented Software Construction*, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[11] Parrish, A., R. Borie, and D. Cordes, "Automated Flow Graph-Based Testing of Object-Oriented Software Modules," *Journal of Systems and Software*, vol. 23, November, 1993, pp. 95-109.

[12] Parrish, A., D. Cordes, and R. Borie, "Developmental Testing of Abstract Data Types," in *Proceedings of the Seventeenth Annual International Computer Software and Applications Conference (COMPSAC)*, Phoenix, AZ, November 1993, pp. 49-55.

[13] Parrish, A., D. Cordes, and R. Borie, "Implementation-Based Testing for Abstract Data Types *Proceedings of the Sixth International Software Quality Conference*, San Francisco, CA., May 1993.

[14] Parrish, A., R. Borie and D. Cordes, "Application and Implementation of Testing Criteria to Reusable Software Modules," *Proceedings of the 31st ACM Southeast Conference*, Birmingham, AL, April 1993, pp. 383-386.

[15] "Catalog of Reusable Software Components," RAPID Center Library, 1991.

[16] "Certification of Reusable Software Components," Digital Systems Research/Rome Laboratory, Final Report, February, 1993.

[17] Spivey, J.M. "The Z Notation: A Reference Manual," Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[18] Weide, B., W. Ogden, S. Zweben. "Reusable Software Components," in *Advances in Computers*, M.C. Yovits, ed., Academic Press, Vol. 33, 1991, pp. 1-65.

[19] Weyuker, E. "The Applicability of Program Schema Results to Programs," *International Journal of Computer and Information Science*, Vol. 8, October 1979, pp. 387-403.

[20] Zweben, S., W. Heym, and J. Kimmich, "Systematic Testing of Data Abstractions Based on Software Specifications," *Journal of Software Testing, Verification and Reliability*, Vol. 1 No. 4, Jan-Mar 1992, pp. 39-55.

MODELOCKED FIBER RING LASER OSCILLATORS AT 1.3 AND 1.55 MICRONS FOR OPTICAL COMMUNICATIONS

Salahuddin Qazi
Associate Professor
Department of Electrical Engineering Technology

State University of New York Institute of Tehnology
P.O. Box 3050, Marcy Campus
Utica, New York 13504-3050

Final Report for:
Summer Research Extension Program
Photonics Laboratory of Griffiss Air Force Base
Rome, New York

Sponsored by:

Air Force Office of Scientific Research
Boilling Air Force Base, Washington, D.C.

and

State University of New York Institute of Technology
Utica, New York.

December 1993

MODE LOCKED FIBER LASER OSCILLATORS AT 1.3 AND 1.3 MICRONS FOR OPTICAL COMMUNICATIONS

Salahuddin Qazi
Associate Professor
Department of Electrical Engineering Technology
State University of New York Institute of Technology

ABSTRACT

This report describes the development of a simple self starting passively mode locked laser oscillator using erbium doped fiber amplifier and low cost modular components. The fiber oscillator generates stable pulses of 1.2 nanosecond width and rates of 5.0 MHz at wavelength around 1.55 micron at pump power as low as 8.0 mW. Progress on the development of fiber laser oscillator at 1.3 micron, based on praseodymium doped fiber amplifier using a laser diode pump of 100 mW output power operating at 1.02 micron, is also reported.

MODELOCKED FIBER RING LASER OSCILLATORS AT 1.3 AND 1.55 MICRONS FOR OPTICAL COMMUNICATIONS

Salahuddin Qazi

INTRODUCTION

The fiber laser as an oscillator has become an important device in all optical fiber communication systems. Modelocked fiber lasers at 1.55 micron wavelength have been recently developed (1,2) to produce short pulses for soliton applications in long haul fiber optic communications. These lasers are designed to produce pulses of picosecond width and maintain their duration throughout the length of fiber in long haul communication systems. Such lasers are in general constructed with specialized components and amplifiers using high pump power and are not self starting. In distributed optical communications and control applications, the use of optical solitons is not very critical and pulses of up to nanosecond width can be tolerated (3).

Comparatively less progress has been achieved in the realization of a viable fiber amplifier or oscillator at 1.3 micron where most of the currently installed optical fiber communication system operate. Praseodymium doped fiber amplifier (PDFA) has been intensively studied and it is the most promising candidate for the 1.3 micron band. This amplifier offers the potential of high signal gain, wide amplified bandwidth and high saturation output power(4). The basic problem (5) of praseodymium doped fiber amplifiers is the low quantum efficiency due to a large percentage of

nonradiative decays from the upper laser level, hence requiring high power pump lasers for amplification. In addition praseodymium fibers are difficult to work with as they are brittle and also cannot be fusion spliced. For fiber optic communications, it is vitally important to develop practical PDFA pumped by laser diodes. In order to use such diodes as pump sources, it is essential to meet the following goals (6).

1. High output power of laser diode pump at 1.017 to 1.02 microns
2. Low coupling loss between laser diode and the doped fiber
3. Highly efficient praseodymium doped fluoride fiber
4. Optimal PDFA and laser configuration

Various PDFA have recently been developed (7) with a gain of over 20 dB by improving the gain characteristics of the doped fibers and output power of pump laser diodes like the use of 4 laser diode pumps and 1 laser diode pump configurations. A recent paper reported (8) the development of a praseodymium doped fiber laser using a non-linear amplifying loop mirror and a Ti:sapphire laser as a pump source.

This report describes the development and characterization of modelocked erbium doped fiber laser as an oscillator. The report also describes the development, currently in progress, of a praseodymium doped fiber laser oscillator at 1.3 micron using one laser diode pump operating at 1.02 micron wavelength. Both of these devices use ring configuration for the oscillators and low power, readily available modular components.

METHODOLOGY

The first part of the project involved the development of erbium doped fiber laser in a unidirectional ring configuration as shown in figure 1. It consists of an erbium doped fiber amplifier, the output of which is coupled to its input through an all fiber coupler, a polarization controller, and a pigtailed isolator of the Faraday rotator type. These components are fusion spliced to reduce the losses and minimize etalon effects which apparently reduce the bandwidth available for modelocking (9). Two erbium doped fiber amplifiers were tried as a gain medium in the fiber laser. The first amplifier was built in the lab and consisted of a length of erbium doped fiber of about 5 micron core diameter (supplied by GTE) pumped with a 100 mW diode laser (SDL- 6312-H1) operating at 980 nm. A fiber wavelength division multiplexer (WDM) having a core diameter of 8 micron was used to combine the pump output with the signal fed back from the amplifier output. Since the pump source is not pigtailed, the output from the pump diode was collimated with a 20x microscope objective and focused onto the input fiber of the WDM with a 5x objective. This launch system coupled 45 mW of pump power at the output of the WDM with a significant loss in pump power occurring at the splice between the WDM output and the erbium doped fiber. The measured loss due to mismatch in fiber diameters at this junction in a good splice was found to be about 1.25 db.

The second amplifier used as a gain medium in the fiber laser was a commercially available optical amplifier(FiberGain Module

model P3-35) of Corning Inc. of New York. It consisted of pigtailed laser diode pump operating at 980 nm, erbium doped fiber and the WDM all included and encased in one module. Such an amplifier provided stable high gain at minimum pump powers. Similar results were obtained by using inhouse built amplifier. The use of Corning FiberGain amplifier eliminated some of the problems described above in case of the inhouse built amplifier. The ring cavity of the fiber laser was constructed by using standard single mode telecommunication fiber with minimum dispersion at a wavelength of 1.3 micron and other readily available components as described above. Various splitting ratios between the output signal and the feedback signal were tried by using different couplers. In the case of higher gain Corning FiberGain amplifier, a ratio of 50/50 gave the maximum output power. For the inhouse constructed amplifier, a 90/10 coupler with 90% of the output fed back to compensate for the lower gain was used.

In order to achieve stable operation of modelocked oscillator an isolator and polarization controller is used in the ring configuration as shown in figure 1. Two types of isolators consisting of pigtailed Faraday rotators were used. The first was non polarizing type (OFR 10-F-IR2), while the second(OFR 10-G-IR2) included a polarizer, hence acting as a polarizing element in the fiber laser ring cavity.

More time was spent in the development of a praseodymium doped fiber laser because of the lack of availability of the doped fibers and high power laser diode pumps operating at 1.017-1.02 micron

range. In addition great care was needed to work with praseodymium doped fibers, which are highly brittle and expensive.

There are only two known vendors of praseodymium doped fibers and one vendor to date for laser diode. Due to budget constraint four meters of praseodymium fiber of 1000 ppm and a core/clad diameter of 6.5/125 microns costing about $4000.00 from LE VERRE FLUORE were purchased. A non-pigtailed laser diode (SDL-6312-H1) of 100 mW output power operating at 1.02 micron for the pump was bought from Spectra Diode Labs of San Jose. This laser diode was arranged to be pigtailed from OZ Optics Ltd. of Ontario, which turned out to be much cheaper compared to a pigtailed laser diode sold by the manufacturer. As it is difficult to fusion splice praseodymium doped fiber, mechanical splicing using V groove was tried as suggested in the literature.

Due to lack of precision tools needed to implement V groove splicing and lack of its flexibility, it was decided to terminate the fiber with a specialized low insertion loss diamond based FC-APC (Angled Physical Contact) connector. Laser diode was pigtailed to a single mode fiber having a core of 6 micron diameter resulting into a 40-52% coupling efficiency. Diamond FC-APC bulkhead adaptor was used to connect the terminated fiber with a single mode glass fiber of 8 micron core diameter, which was then fusion spliced to the WDM.

MEASUREMENT APPARATUS

In order to optimize the performance of the developed fiber

laser oscillator and determine the physical mechanism for their behavior, optical spectrum and the temporal characteristics of the output was examined. The optical spectra were obtained with a commercial optical spectrum analyzer (Anritzu MS900 1B1) and the temporal characteristics were observed using two combinations. The first combination consisted of a fast photodiode amplifier (Lasertron QRX-700) and a digitizing signal analyzer (Tektronix DSA 602) and the second combination consisted of an avalanche photodiode (Antel AMF-20, Mod..MARG-20) and a digitizing sampling oscilloscope (Tektronix 11801). The first combination featured high sensitivity and rapid response but had a band pass of 500 MHz. The band pass of the second combination was measured to be 7.54 GHz corresponding to a resolution of 133 ps. In addition to the above measuring instruments , an optical autocorrelator (INRAD 5-14-AD) was also used to detect structure faster than the resolution of the above detection systems.

RESULTS

The end product of the work to date is two erbium doped fiber laser oscillators, one based on Corning FiberGain amplifier and the other on the inhouse built amplifier. Work is in progress on the investigation of these devices and implementation of praseodymium doped fiber laser oscillator. The following results were obtained from the fiber laser based on Corning FiberGain amplifier, although results obtained from the fiber laser based on the amplifier constructed in house were very similar.

In order to characterize these devices, optical spectrum and the

temporal behavior of the fiber laser output were examined as a function of pump power. Figure 2 shows an oscilloscope trace of time averaged output of the oscillator near the threshold for the lasing action. It was found that the oscillator's output always consisted of CW components and a periodic series of short pulses at the threshold level and above. No special effort was needed to initiate the short pulse component of the output. The period of the pulses was remarkably stable and independent of pump power, enabling averaging techniques to be used to accurately measure the pulse period and average width. The pulse period was found to be 189.2 ns, which corresponds to a frequency of 5.285 Mhz. The average pulse width was measured using a detection system with a bandwidth of 7.55 GHz, and was found to be 1.246 ns.

The period of the output pulses was measured as a function of the length of the oscillator cavity and the result is shown in figure 3. The results were obtained by fusion splicing known lengths of standard telecommunications fiber in the feedback loop forming the cavity. It is clear from the result that the pulse period was proportional to the cavity length. The length of the fiber used in these measurements ranged from 46 meters to 56 meters.
A large fraction of the total cavity length was made up of the erbium doped fiber in the amplifier, which was estimated to be approximately 22 meters in length.

The time averaged output of the oscillator was measured as a function of launched pump power using a standard power meter. The launched pump power at the threshold for oscillation was found to

be about 4 mW, while the output power was 3 mW for a pump of 25 mW.

The optical spectrum of the oscillator was also measured as a function of pump power. The result in Figure 4 shows that near the threshold the spectrum consisted of a band with a full width at maximum of about 2 nm centered at 1.556 nm. As the pump power was increased, oscillation occurred in several narrower bands as illustrated in figure 5. The central band shown here had a half width of about 0.5 nm.

The temporal behavior of the oscillator output was also analyzed with an autocorrelator. The result is shown in figure 6 for the same pump power used to obtain the data presented in figure 5. The presence of temporal pulses with a full width at half maximum of about 40 ps is indicated by this data. At pump powers near the threshold the output power of the oscillator was too low for autocorrelation measurements to be made.

In the development of praseodymium doped amplifier, 52 mW was coupled to the fiber coupler from 100 mW pigtailed pump laser diode at the operating current of 130 mA. Total loss with three meters praseodymium fiber terminated with FC-APC connectors and joined to 1.5 meter single mode glass fiber by diamond FC-APC bulkhead adaptor on each side was tested to be about 2.00 dBs.

DISCUSSION AND CONCLUSIONS

The appearance of a stable train of short pulses with a period set by the length of the oscillator cavity indicates that some form of self mode locking was occurring in the laser oscillator described above. This is supported by the presence of structure on

a picosecond scale in addition to the broader nanosecond pulses. A comparison of Figures 5 and 6 indicates that this picosecond structure was band width limited. It is possible that the observed nanosecond pulses contained unresolved trains of picosecond pulses. This behavior has been observed to occur in other types of erbium doped fiber oscillators (10). The broad spectral band observed near threshold in figure 4 indicates the possibility of bandwidth limited structure as narrow as 2 ps. It is worth emphasizing that the mode locking of this laser oscillator was self starting. In fact mode locking seems to be the preferred method of oscillation and persists over the entire range of pump powers investigated. It should also be noted that the pump powers used in this investigation were quite low compared to those reported in other work.

In summary the ring laser oscillator developed in this project, demonstrates the simplest, passive configuration with the lowest threshold to exhibit self pulsed effects. Most other laser oscillator configurations reported to date, require additional components such as modulators and etalons. This system design is modular and shows significant progress toward the goal of a practical pulsed or CW fiber laser. Due to the delay in acquiring suitable components, more time is needed to obtain results from the developed praseodymium doped fiber laser which remains to be an important device for the existing fiber optic telecommunication systems.

ACKNOWLEDGEMENTS

The author would like to acknowledge the assistance of Photonics Center of Rome Laboratories in providing experimental facilities to complete the work. The support of the Office of Scientific Research of the United States Air Force and the Research Foundation of State University of New York through the office of Sponsored Research at the SUNY Institute of Technology is also greatly acknowledged.

REFERENCES

1. E.P. Ippen, H.A. Haus, and L.Y. Liu, J. Optical Society of America, Volume B6,PP, 1736 , 1989.

2. N.J. Doran and D. Wood, Optics letters, Volume 13, PP,56 ,1988.

3. K.J. Teegarden, R.K. Erdmann, S. Qazi, International Society of Engineering Symposium OE/Fibers, Boston, 1992.

4. Y. Ohishi et.al., Electronic Letters, Volume 27,PP,1706-1707, 1991.

5. Anders Bjarklev, Optical Fiber Amplifiers, Artech House, pp 277, 1993.

6. Makato Yamata et. al.,Optical Amplifiers And Their Applications, PP 240, Yokohoma, Japan, 1993.

7. M. Shimuzu et al., OFC 93, San Hose, PD12-1, pp, 52-55, 1993.

8. T.Sugawa et al., Optical Amplifiers And Their Applications, TuA6-1, PP 260, Yokohama, Japan, 1993.

9. H.A. Haus and E.P. Ippen, Optics Letters, Volume 17, PP 1331, 1991.

10. D.J. Richardson, R.I. Laming, D.N. Payne, et al., Electronic letters, Volume 27, PP 1451, 1991.

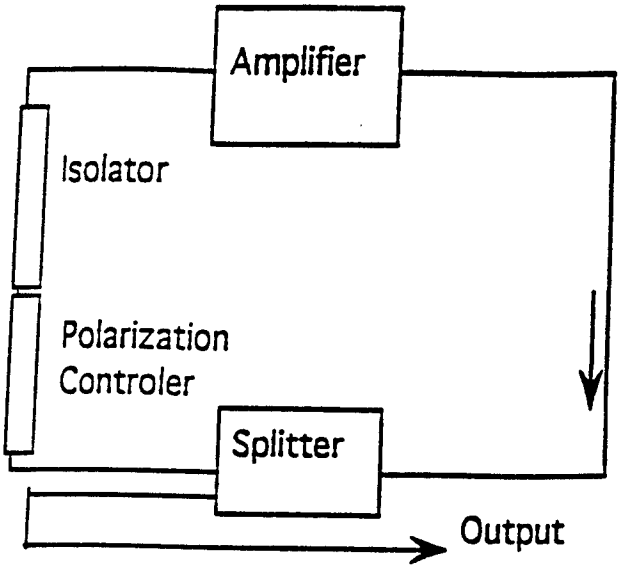


Fig. 1 Oscillator layout

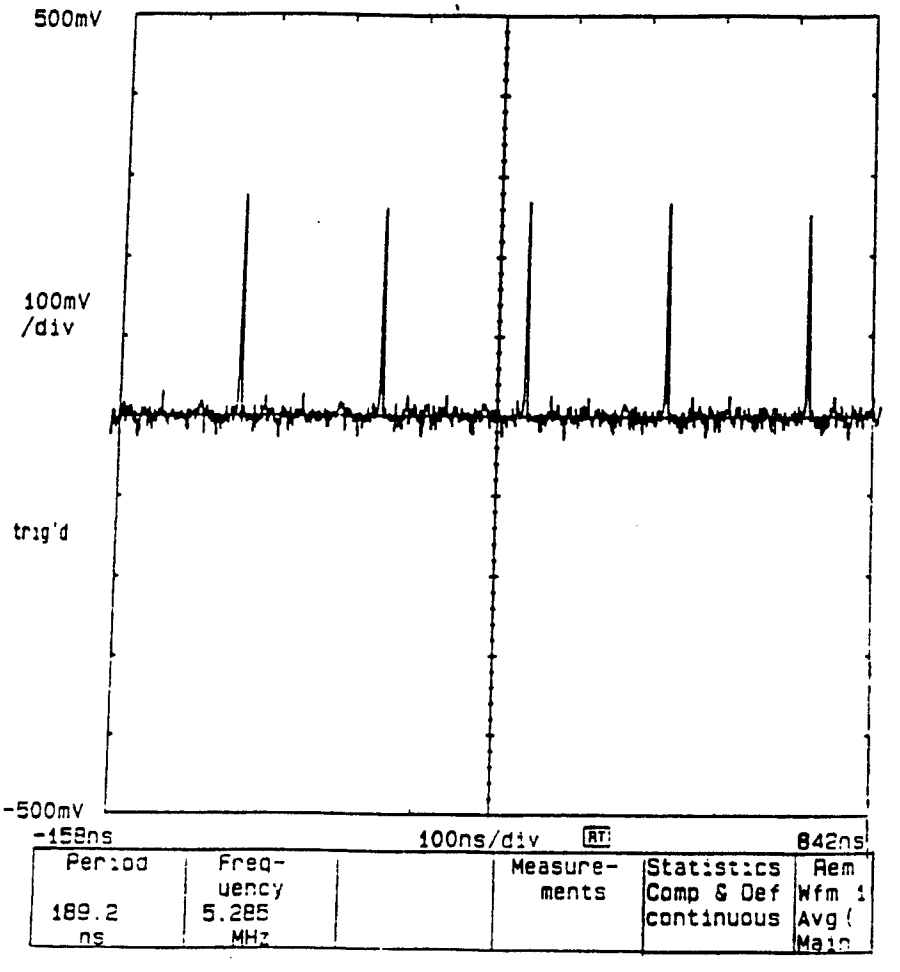


Fig. 2 Time average of oscillator output at 8.0 mW pump power.

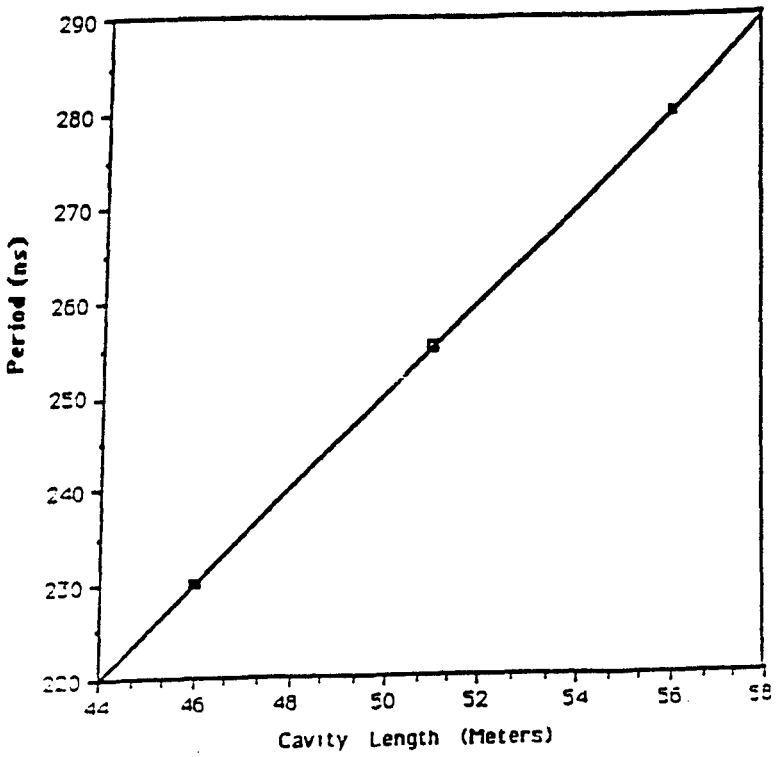


Fig. 3. Period of output pulses as a function of cavity length

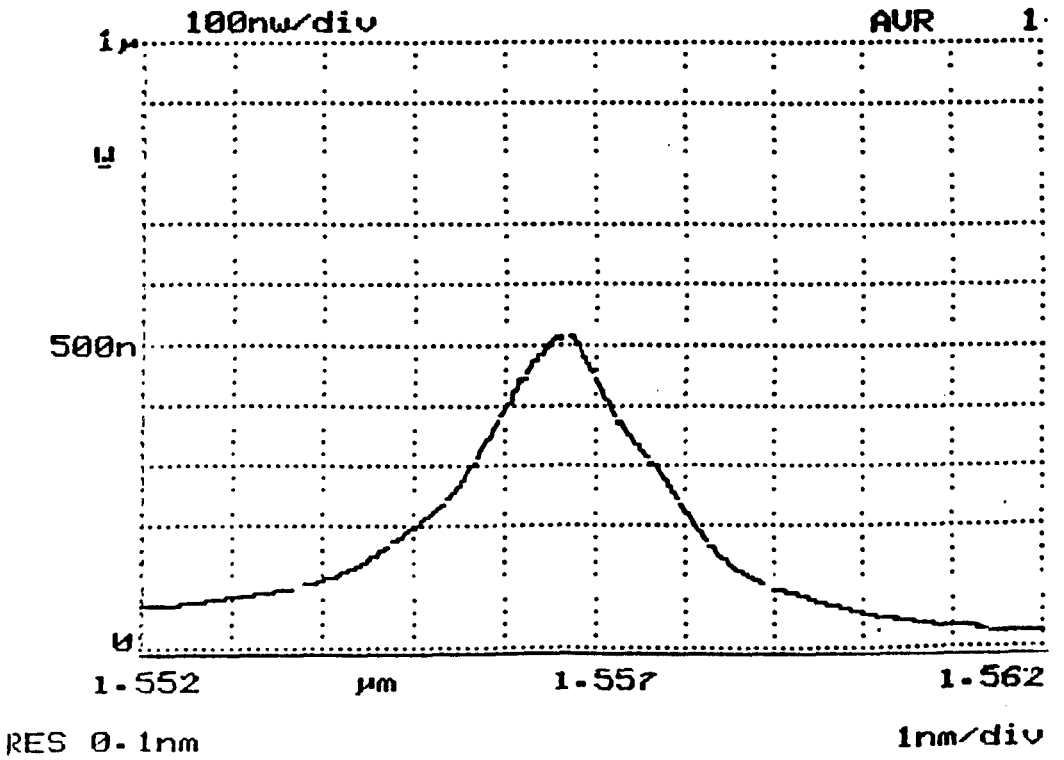


Fig. 4. Optical spectrum near threshold.

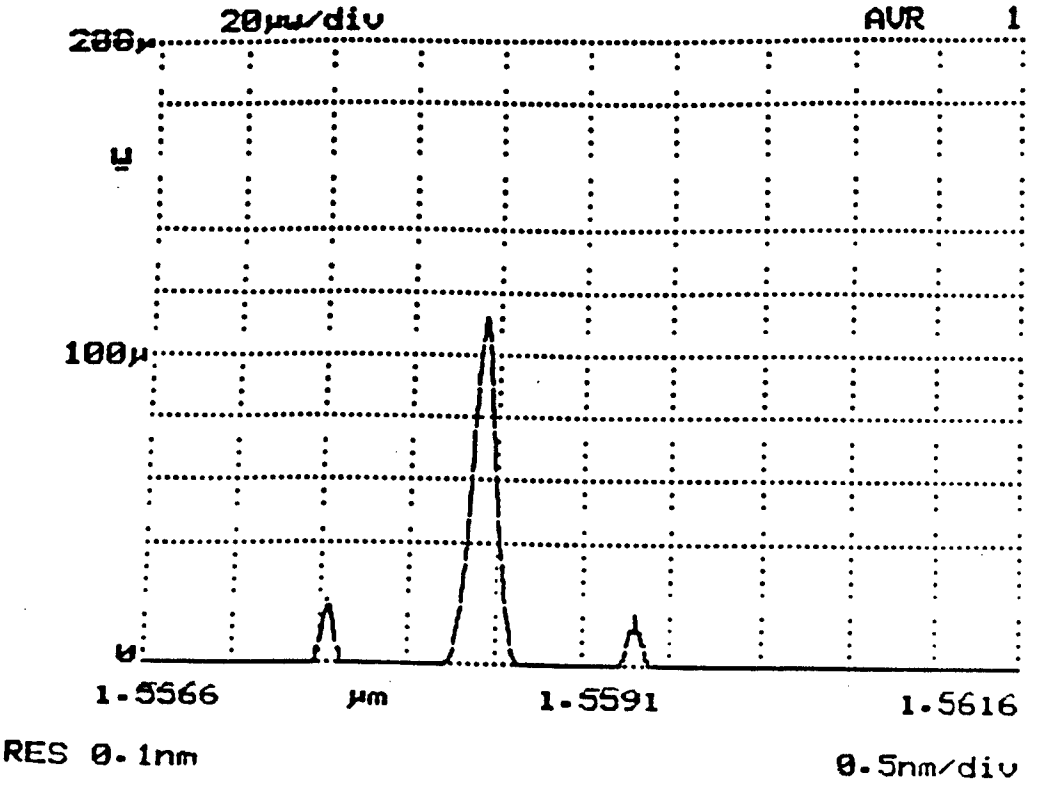


Fig. 5. Optical spectrum at higher pump powers.

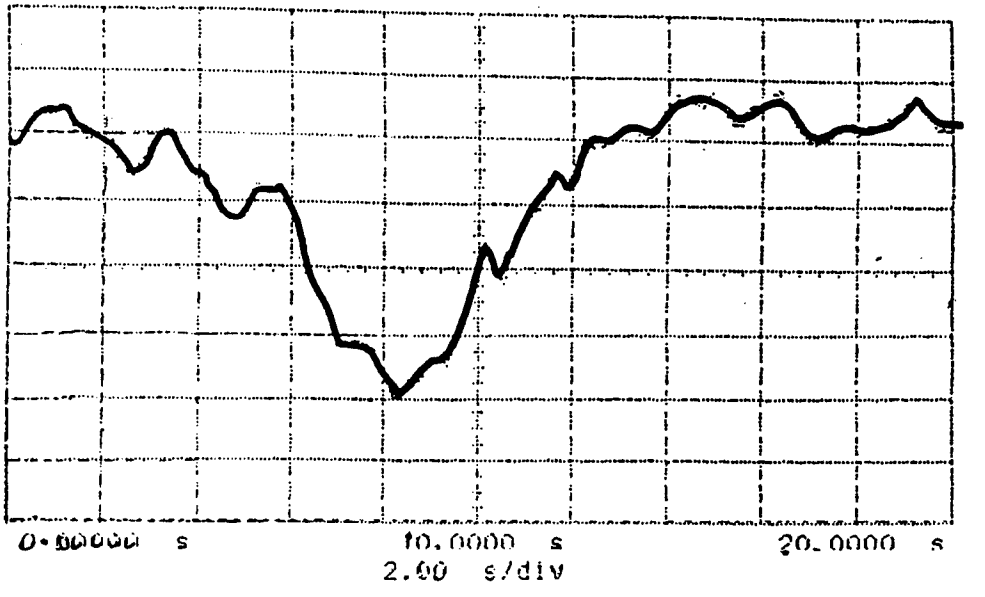


Fig. 6 Autocorrelation trace of output at 20 mW pump power.
One major division equals 20 ps.

FURTHER MONTE CARLO STUDIES OF A THEORETICAL MODEL FOR NON GAUSSIAN RADAR CLUTTER CHARACTERIZATION.

JORGE LUIS ROMEU
ASSOCIATE PROFESSOR
DEPARTMENT OF MATHEMATICS

STATE UNIVERSITY OF NEW YORK
COLLEGE AT CORTLAND
CORTLAND, NY 13045

FINAL REPORT
SUMMER RESEARCH INITIATIVE PROGRAM
ROME LABORATORY

SPONSORED BY:
AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
BOLLING AIR FORCE BASE, WASHINGTON, D.C.

NOVEMBER 15, 1993

FURTHER MONTE CARLO STUDIES OF A THEORETICAL MODEL FOR NON GAUSSIAN RADAR CLUTTER CHARACTERIZATION.

JORGE LUIS ROMEU
ASSOCIATE PROFESSOR
DEPARTMENT OF MATHEMATICS
SUNY CORTLAND
RESEARCH FELLOW OF THE
CASE CENTER OF SYRACUSE UNIVERSITY

ABSTRACT

The present Summer Research Initiative Program (SRIP) project continues and expands the validation scheme in Romeu (1992a, 1992b and 1993a). It builds up on previous work by Rangaswamy, Weiner, Ozturk, Michels and Romeu, and develops a methodology for a general empirical Goodness of Fit (GOF) test for Spherically Invariant Random Processes (SIRP). This testing methodology circumvents having to obtain a mathematically intractable distribution, required for GOF testing of multivariate SIRP's. The exact pdf is substituted by Monte Carlo derived estimators. With such test, any incoming signal (SIRP X) may be identified as one pre-specified. This test approach can also be used to assess and compare different SIRP covariance estimators and to study the effects of different SIRP parameters and characteristics in the estimation process.

This final report discusses the three phases of this study. Phases I and II describe, respectively, the procedures for obtaining the parameters of the GOF test and their validation. Phase III studies the effects of several SIRP parameters on the estimation of the SIRP covariance matrix and on the GOF (identification) test. Future Research and other potential uses of this GOF test approach to the identification of an incoming signal are also provided.

FURTHER MONTE CARLO STUDIES OF A THEORETICAL MODEL FOR NON GAUSSIAN RADAR CLUTTER CHARACTERIZATION.

Jorge Luis Romeu

INTRODUCTION.

This research continues and expands the validation approach started in Romeu (1992a, 1992b and 1993a), according to the roadmap proposed in Romeu (1992c). In Romeu (1992a), a Monte Carlo methodology to validate the generation of several special cases of Spherically Invariant Random Process (SIRP) X, (i.e. the Gaussian and the univariate K-Distributed SIRP's) was developed. And in Romeu (1992c), it was proposed to extend this methodology, through a modified Goodness-of-Fit (GOF) test approach, to study several multivariate characteristics and parameters of Spherically Invariant Random Process (SIRP).

In the present report, we implement that proposal. We complete the validation scheme in Romeu (1992a) by implementing an empirical test for the general K-Distributed case, when the number of variables (N) of the SIRP X (e.g. $X' = (X_1, \ldots, X_N)$ is greater than one (e.g. $N > 1$) and are **inter-correlated** (as opposed to independent). And a methodology for the comparison of various estimation procedures for the covariance matrix, using the GOF test as a tool, is developed.

It is theoretically known (Rangaswamy (1992), Rangaswamy et al. (1991, 1992), Kaman (1992)), that the density $f_P(.)$ of the quadratic form $p = X'\Sigma^{-1}X$, of an SIRP $X = s * Z$ (where s is the process driver, Z is multivariate Gaussian with covariance Σ and N is an integer) is given by:

$$f_P(p) = \frac{1}{2^{N/2}\Gamma(N/2)} p^{N/2-1} h_N(p)$$

$$where \qquad h_N(p) = \int_0^\infty s^{-N} exp(\frac{-p}{2s^2}) f_S(s) ds$$

However, for $N > 1$, the expression for $h_N(p)$ typically does not have a mathematically simple form. For example, some pdf's of interest incorporate modified Bessel functions. The corresponding $f_P(p)$ are difficult to obtain, analytically, even after performing approximations, changes in variables, and other mathematical manipulations. Therefore, taking $f_P(*)$ to a form that can be programmed may require a considerable amount of time and effort. And, even after all this analytical work, the accuracy of the result may be questionable.

On the other hand, we can make full use of the Ozturk approach to distribution identification (see Section 6.4 of Kaman (1992)). As stated there, the approach uses the Ozturk univariate Q_n GOF test (Ozturk and Dudewicz (1990)) on the quadratic form p to provide an indication about

what is the true distribution of p. Such indication depends on the location of the Q_n statistic with respect to the distribution approximation chart. The SIRP X is then identified by choosing the most plausible distribution (for the quadratic function p) among all candidates in the distribution approximation chart.

Our proposed approach is a modification of the Ozturk one and complements it in several ways. First, our approach modifies the univariate Q_n statistic by using a different **reference distribution** than Gaussian. Then, it assumes that the (candidate) SIRP distribution is known (i.e. has been established by selecting, in the approximation chart, the quadratic form p that characterizes it). Finally, our proposed approach tests, via the (univariate) quadratic function p of the SIRP X, whether or not the distribution identification is correct. Hence, it provides a probabilistic statement about how good (or poor) this identification is.

In addition, one can use our proposed GOF testing approach in two other areas. First, as an assessment tool to evaluate different factors that affect the behavior of an SIRP. And then, to compare the performance of different SIRP covariance matrix estimators. Specifically, one can evaluate different estimators Σ^* of the true (and unknown) covariance matrix Σ of the SIRP X. And one can investigate the effects of (i) the sample size n, (ii) the number of variates N (vector size), (iii) its inter-correlation ρ, that we will use to specify the SIRP (experimental) covariance matrix:

$$\Sigma = \begin{pmatrix} 1 & \rho & \dots & \rho \\ \rho & 1 & \rho & \dots \\ \rho & \dots & \rho & 1 \end{pmatrix}$$

and (iv) the interaction among all these factors, through the performance of our **modified** GOF test statistic Q_n.

Finally, by testing the SIRP's using this modified Q_n approach, we complete the SIRP model validation scheme started in Romeu (1992a). There, we studied the SIRP (*i*) for $N = 1$ and (*ii*) for the special (Gaussian) case for $N > 1$. We also assessed via Monte Carlo, the efficiency of statistics p and p^*, obtained with the true or estimated covariance matrix. In the present research we complete that work. First, by studying (*iii*) the special case of a Bi-dimensional K-Distributed SIRP X. This case has a closed form solution to compare with. Then, we study (*iv*) the general ($N > 1$) case.

However, our proposed use of Ozturk's approach (i.e. to perform a GOF test on the quadratic form p) requires the distribution function F_P and density function f_P of p. And these are extremely difficult to evaluate numerically in the general case. We have circumvented this problem by obtaining, via Monte Carlo (Kennedy and Gentle (1992)) the empirical distribution of the quadratic form p and of Ozturk's univariate statistic $Q_n = (U_n, V_n)$, for a specific *setting* (n, N, ρ), given by the sample size, the vector size and the vector inter-correlation.

We propose that the univariate Q_n be modified by using a reference distribution other that Gaussian (we will use the specific distribution of the quadratic form p, in each case). We use this

modified Q_n to test the GOF of the quadratic form p obtained from a pre-specified SIRP X, for the given setting (n, N, ρ).

An additional use of our proposed testing approach is as an assessment tool. In this case we perform Monte Carlo experiments, sampling from the same SIRP process X (i.e. with the same sample size n, the same vector size N and the same covariance matrix Σ as those used to obtain the corresponding empirical distribution of Q_n). But now we use the modified Q_n GOF test results to assess possible effects of the different combinationf of (n, N, ρ) under study.

We can also use our proposed approach to assess the efficiency of $p* = X'\Sigma^{*-1}X$. Here p^* is the estimated quadratic form obtained by using different covariance matrix extimators Σ^*. Using the proposed approach we interpret the results of the modified Q_n GOF test as an assessment of the effects of the mentioned SIRP factors on the performance of the covariance matrix estimators Σ^* employed.

In the following sections of this report we discuss the three sequential phases of our research:

(*i*) Estimation of the Empirical Distribution of Q_n

(*ii*) Validation of the Empirical Parameters Obtained

(*iii*) Evaluation of SIRP Factors and Covariance Estimators

Finally, for a possible *Future Research* phase we briefly discuss the development of a Research and Classification Parameter Bank and other ideas. In such future extension, we propose that specific clutter scenarios (e.g. forest, hills) be characterized by specific K-Distributed SIRP processes X. These, in turn, can also be mapped into their corresponding quadratic functions p. Then, having a Bank of Distribution Parameters for these specific quadratic forms p (via Monte Carlo) would provide another means for the rapid classification of an incoming radar signal. For, the incoming quadratic form could be approximated to one of the pre-existing ones and the clutter scenario that it characterizes could be identified.

The above described summer research program has been ambitious and intensive. We implemented an experimental design of settings (n, N, ρ), for a K-Distributed SIRP process $X = s * Z$. However, since we developed new methodology in very limited time, we were only able study a limited amount of these settings. We look forward to further work in a future project extension.

PHASE I: ESTIMATION OF THE EMPIRICAL DISTRIBUTION OF Q_n.

In this phase we obtain, via Monte Carlo, all necessary parameters to carry out our modification of the Q_n test for the quadratic form p. Such parameters are obtained as explained below.

In order to establish the experimental settings (n, N, ρ) for our research (Cochran and Cox (1992)) we need to perform a theoretical analysis of the candidate SIRP parameters. This is done under the light of (*i*) the SIRP theory and (*ii*) the above mentioned research objectives. We thus define

convenient (research) parameter values for (i) the vector size N, the (ii) number of independent samples n and the (iii) covariance matrix Σ, (of a K-Distributed SIRP process X) defined via the vector's inter-correlation ρ.

Our analysis yielded the conveniently tractable special case of the K-Distributed bivariate ($N = 2$) SIRP $X = s * Z$, with shape parameter $\alpha = N/2 - 0.5$ and scale parameter $b = 1$. This special SIRP has a quadratic form p with closed form pdf function $f_P(*)$:

$$Given \quad h_N(p) = \frac{\sqrt{p}^{N/2-0.5-N/2}}{\Gamma(N/2-0.5)2^{N/2-0.5-1}} \times K_{N/2-N/2+0.5}(\sqrt{p}) = \sqrt{\frac{\pi}{p}}\frac{exp(-\sqrt{p})}{\Gamma(\frac{N-1}{2})2^{\frac{N-2}{2}}}$$

$$and \quad K_{\frac{N}{2}-\alpha} = K_{0.5}(x) = K_{-0.5}(x) = \sqrt{\frac{\pi}{2}} \times \sqrt{\frac{1}{x}}exp(-x)$$

$$Then \quad f_P(p) = \frac{1}{2^1\Gamma(1)} \times p^{1-1}h_2(p) = \frac{1}{2}p^{-1/2}exp(-\sqrt{p})$$

These functions will provide the theoretical comparison values to validate our Monte Carlo study of the SIRP process $X = s * Z$. However, in the above form, this pdf is still too complex for us to work with, directly. And certain transformations are required to simplify our work.

Under the transformation $\frac{w^2}{2} = \sqrt{p}$, the resulting random variable (r.v.) w is distributed Rayleigh and easy to test for GOF. Such GOF test is necessary (i) to validate our Monte Carlo experiment and (ii) to compare the Power of our modified Q_n test with an established GOF test for p. It is statistically equivalent to test GOF directly on the r.v. p or on its transformation, the r.v. w. For, if $p \sim F_P$ then $w \sim$ Rayleigh. We select the second alternative for its ease and computational speed.

Accordingly, the pdf $f_S(*)$ of the driver random variable $s > 0$, for this special case of K-Distributed SIRP $X = s * Z$ is:

$$f_S(s) = \frac{2}{\Gamma(\alpha)2^\alpha}s^{2\alpha-1}exp(\frac{-s^2}{2}) = \sqrt{\frac{2}{\pi}}exp(-s^2/2)$$

which, under the transformation $y = s^2/2$ becomes:

$$f_Y(y) = \frac{1}{\sqrt{\pi}}\frac{1}{\sqrt{y}}exp(-y) = \frac{1}{\Gamma(1/2)}y^{\frac{-1}{2}}exp(-y)$$

i.e., a Gamma distribution with parameters $\lambda = 1, r = 1/2$. It is now easier and faster to generate a Gamma r.v. y and obtain $s = \sqrt{2y}$.

The above random variable s also has the convenient property that:

$$E(s^2) = \int_0^\infty s^2\sqrt{\frac{2}{\pi}}exp(-s^2/2)\,ds = 1$$

therefore, insuring that the resulting covariances ($\Sigma = M$) of our Gaussian (Z) and SIRP $X = s * Z$ processes are equal.

We also investigated other special cases of SIRP for $\alpha = N/2 - 0.5$ and $b = 1$:

$$For \quad N = 3 \qquad f_P(p) = \frac{\sqrt{\pi}}{4\Gamma(1.5)} e^{-\sqrt{p}}$$

$$For \quad N = 4 \qquad f_P(p) = \frac{\sqrt{p\pi}}{8\Gamma(2)} e^{-\sqrt{p}}$$

$$For \quad N = 8 \qquad f_P(p) = \frac{p^2\sqrt{p\pi}}{3!2^7\Gamma(3.5)} e^{-\sqrt{p}}$$

none of which yields a well known pdf and all of which exhibit the same numerical difficulties that we are precisely trying to avoid in this research.

We proceed by Monte Carlo to obtain the empirical distribution F_P^* of the quadratic form $p = p_N = X'\Sigma^{-1}X$ (verify how the quadratic form p also depends on the dimensionality N of the covariance matrix Σ). This is necessary because, in general, we lack the theoretical distribution function $F_P(*)$ of the quadratic form p (as illustrated in the examples above, for $N = 3, 4, 8$). However, F_P is completely determined by $s, h_N(*), \Sigma$. Hence, we use this knowledge to obtain, by Monte Carlo, the distribution $F_P(*)$ and use it to implement the modified version of Q_n. This, in turn, allows us to assess whether the quadratic form p really corresponds to the specified $X = s * Z$, or not.

Romeu (1993a) studied (*i*) the special K-Distributed case $N = 1$, and (*ii*) the Gaussian SIRP, for $N > 1$. Romeu showed, for these two cases, how the quadratic form p_N is a very powerful statistic for correctly identifying an SIRP, when Σ is known. And how p_N is still good even when Σ is unknown and estimated from the data, as long as the data set is large enough. Romeu proposed the study of a higher order, more general case of K-Distributed SIRP X, which is the main objective of the present study.

In general (and for $N > 1$) it is very difficult to analytically obtain a closed form for the density (pdf) $f_P(*)$ of p_N of an SIRP X. However, we can approximate its distribution (CDF) $F_P(*)$ via Monte Carlo in the following way:

Let the **modified** $Q_n = (U_n, V_n)$ GOF test, be defined:

$$U_n = \frac{1}{n}\sum_i^n cos\theta_i \, |Z_i|$$

$$V_n = \frac{1}{n}\sum_i^n sin\theta_i \, |Z_i|$$

$$with \qquad \theta_i = \pi \int_{-\infty}^{m_{i:n}} f_P(t)dt = \pi F_P(m_{i:n})$$

where $m_{i:n}$ is the i^{th} order statistic (David (1990)) from the ordered sample of the corresponding n quadratic forms p_N, denoted $p_1 < p_2 < \cdots < p_n$. Let these n samples be obtained from the simulated

SIRP $X = s * Z$. And let F_P, f_P be, respectively, **the distribution and density functions of quadratic form** p. This is the modification of the original Q_n of Ozturk and Dudewicz.

In addition $Z_i = \frac{p_i - p_{avg}}{S_p}$, be the standardized p_i's, where p_{avg} is the average of the n simulated quadratic forms p_i. Finally, let S_p be the corresponding sample standard deviation of the n simulated p_i's.

To obtain the **angles** θ_i, $i = 1, \ldots n$, which yield the endpoints (U_n, V_n) of the modified Ozturk statistic, we require the distribution function F_P. Since, in general, we cannot obtain F_P analytically, we approximate $F_P^*(*)$, via Monte Carlo, in in the following way.

First, we generate NTOT samples of size n each, from the same SIRP X, defined above. For $j = 1, \ldots NTOT$, we then obtain the ordered sample $p_{1,j} < \cdots < p_{n,j}$ of quadratic forms. Finally, we obtain:

$$m_{i:n}^* = \frac{1}{NTOT} \sum_{j}^{NTOT} p_{i,j}, \quad for \quad i = 1, \ldots n$$

From the empirical $m_{i:n}^*$ values, we obtain the empirical cdf's $F_P^*(m_{i:n}^*)$, also by Monte Carlo:

$$F_P^*(m_{i:n}^*) = \frac{pp < m_{i:n}^*}{NTOT}, \quad i = 1, \ldots n$$

where $pp < m_{i:n}^*$ is the number of simulated quadratic forms, out of the total NTOT generated in the Monte Carlo experiment, that are smaller than the corresponding $m_{i:n}^*$, obtained as above.

We thus obtain $F_P^*(*)$ at each of the empirical order statistics $m_{i:n}^*$ of the quadratic form p_N, of a given sample size n. From these values, the empirical angles $\theta_i^* = \pi F_P^*(m_{i:n}^*)$ are easily obtained. But we also have, for the special case of the Bivariate K-Distributed SIRP X (a statistical equivalent of) the exact distribution $F_P(*)$, of the quadratic form p_N. And we can use it to validate our Monte Carlo procedure (Romeu, (1992d)) by comparing how close both of these GOF test results are.

A sample output of the implementation of our approach, showing the Monte Carlo derived $F_P^*, m_{i:n}^*, \theta_i^*$ for $N = 2$, $n = 10$, $\rho = 0.5$, is presented in Table 1.

Table 1. Sample of Empirical Values for $N = 2$, $\rho = 0.5$:

i^{th} Obs	$\lvert\frac{p - p_{avg}}{s}\rvert$	$m_{i:n}^*$	$F_P^*(m_{i:m}^*)$	θ_i^*
1	0.683238	0.019328	0.128210	0.402783
2	0.660789	0.066707	0.224870	0.706450
3	0.621371	0.151513	0.320800	1.007822
4	0.560787	0.284144	0.410180	1.288618
5	0.467949	0.497728	0.501950	1.576921
6	0.342191	0.833601	0.594480	1.867613
7	0.245568	1.381543	0.687350	2.159373
8	0.347593	2.324335	0.778580	2.445980
9	0.869920	4.265456	0.871280	2.737206
10	2.337782	10.124611	0.957710	3.008734

Using these values we obtain, through a second Monte Carlo experiment, the empirical estimators of the parameters $E(U_n), E(V_n), \sigma_u^2, \sigma_v^2, \rho_{uv}$, required for implementing the modified Q_n GOF test. A sample output of these empirical estimators, corresponding to the values obtained in Table 1, is presented in Table 2. The theoretical value for $Q_n = (U_n, V_n)$ is given for comparison.

Table 2. Corresponding Simulation Results for $N = 2$, $\rho = 0.5$:

Q_n	THEORY	MEAN	VARIANCE	CORREL.
U_n	-0.200407	-0.199829	0.001842	0.189824
V_n	0.363497	0.362216	0.005134	0.189824

Notice how the simulated results for the mean of the modified GOF test $Q_n = (U_n, V_n)$ are within three standard deviations of their corresponding theoretical values, even for this small sample size example($n = 10$).

The set of empirical values $F_P^*(*), m_{i:n}^*, \theta_i^*$, $i = 1 \ldots n$ (of Table 1) are calculated only once for each parameter setting (n, N, ρ), via a Monte Carlo experiment. They are then repeatedly used, in subsequent phases of this study, for testing the GOF of the SIRP X (via the GOF test of quadratic form p). We, thus, need to validate these empirical parameters before using them in the proposed testing methodology.

PHASE II: VALIDATION OF THE EMPIRICAL DISTRIBUTION OF Q_n.

In the previous phase (Phase I) $NTOT = 10,000$ samples of size $n = 25, 50, 100, 200$ of an N-variate SIRP X were generated on Syracuse University's IBM 3090, using the IMSL statistical library. We used vector size values of $N = 2, 4, 8$ with covariance matrix Σ defined as above explained, for $\rho = 0.0, 0.5, 0.9$. The experiments for $N = 4, 8$ were implemented at the Cornell Supercomputer (vector) Facility, given their extensive run times in a sequential computer such as SU's IBM 3090.

For each simulated sample (i.e. each simulated setting (n, N, ρ)) we calculated the quadratic forms p_N of the SIRP X and obtained the **empirical estimations** of (i) the corresponding order statistics $m_{i:n}$, (ii) distributions $F_P(m_{i:n})$, and (iii) angles θ_i, for $i = 1, \ldots n$. These three sets of estimations $(m_{i:n}^*, F_P^*, (m_{i:m}^*), \theta_i^*)$ in turn allowed the calculation of NTOT Monte Carlo $Q_n^* = (U_n^*, V_n^*)$ statistics. Finally, these Q_n^* values provided the empirical mean, variance and correlation of the bivariate distribution of Q_n. With them, we derived the confidence ellipsoids for our GOF tests.

The *modified* Q_n procedure uses, as reference distribution, that of the quadratic form p_N (and not that of the Normal distribution, as in the original work of Ozturk and Dudewicz). This allows testing the GOF of the (multivariate) SIRP X (i.e., for $N > 1$) using the (univariate) quadratic form p_N. In this way we avoid having to resort to a (rarely available) multivariate GOF test for X.

The present Phase II **assesses the empirical estimations** of the $F_P(*), m_{i:n}, \theta_i$, $i = 1, \ldots, n$, and of the mean, variance and correlation of the bivariate distribution of Q_n. The GOF statistic Q_n is now obtained with reference to F_P, the distribution of quadratic form p_N. We assess the validity of our proposed statistic Q_n by comparing the Monte Carlo derived Q_n^* GOF test results with those from the special case of the Bivariate K-Distributed SIRP X (with shape parameter $\alpha = \frac{N}{2} - 0.5$). Recall how this exact distribution was already obtained in Phase I. Since both of these GOF tests assess the same SIRP Process X, both results should agree. And since we are simulating from (H_0) the same Bivariate K SIRP, these two GOF test results should also be close to their nominal significance levels.

By generating from the same SIRP X used in Phase I we obtain, based on Johnson and Kotz (1970), that the statistic $Q_n = (U_n, V_n)$ fulfills, approximately:

$$\frac{1}{1-\rho_{uv}^2}\{\frac{(U_n - E(U_n))^2}{\sigma_u^2} - 2\rho_{uv}\frac{(U_n - E(U_n))(V_n - E(V_n))}{\sigma_u\sigma_v} + \frac{(V_n - E(V_n))^2}{\sigma_v^2}\} \quad \sim \quad \chi_2^2$$

With this equation we obtain the confidence ellipsoids to implement the empirical GOF tests using the statistic $Q_n^* = (U_n^*, V_n^*)$. Examples of the bivariate distributions of statistic Q_n^*, obtained with 5000 replications for $n = 10, 50, 200$ and $N = 2$, are shown in Figures 1, 2 and 3. The confidence ellipsoids are approximations to **slices** from such bivariate distributions.

We obtain, by Monte Carlo, estimations of the expected values and variances of $Q_n = (U_n, V_n)$ (i.e. $E(U_n), E(V_n), \sigma_u^2, \sigma_v^2$) and of their intercorrelation (ρ_{uv}). And using the above equation with Monte Carlo parameter estimations, we obtain the desired confidence ellipsoids.

We then use these confidence ellipsoids to test each simulated sample. We use the modified Q_n GOF test statistic in the following way: (i) accept H_0 (i.e. that the distribution of p_N is as stated) whenever Q_n^* falls within the ellipsoid, and (ii) reject H_0 otherwise.

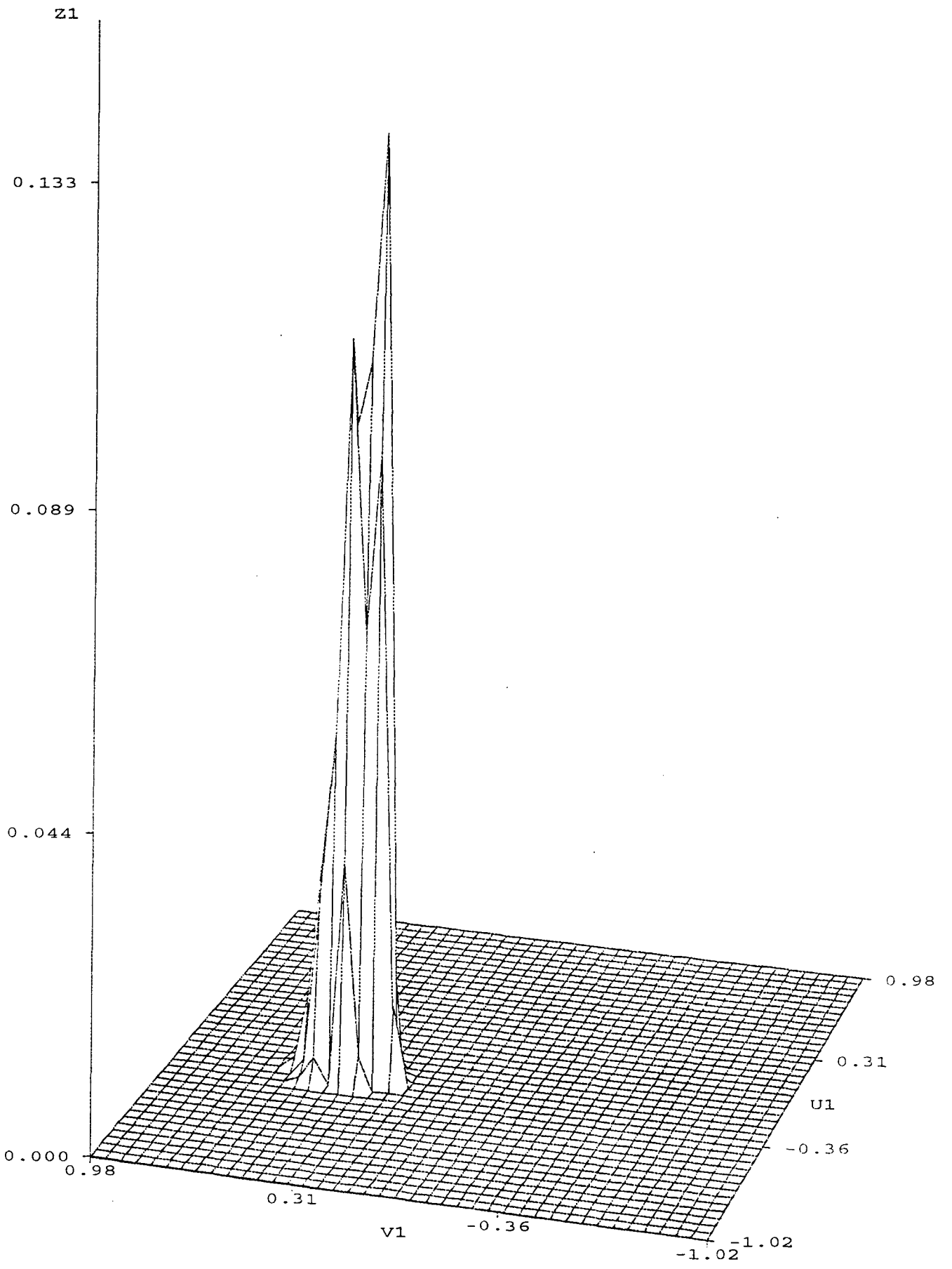
The results of this modified Q_n GOF test are compared with those of the **Exact** GOF test. The **Exact** test is performed on the transformation w of quadratic form p_N (obtained during Phase I) for each generated sample of our Monte Carlo experiment.

Using this special Bivariate K SIRP, we validate our proposed testing approach in two ways. First, we obtain the **percent rejections** (P_r) for both the modified Q_n and Exact GOF tests, at the usual significance levels of $\alpha = 0.1, 0.05, 0.01$. And we assess whether they are within three standard deviations of the nominal α's:

$$\alpha \quad (+/-) \quad 3\sigma_\alpha(NTOT) = \alpha \quad (+/-) \quad 3 \times \sqrt{\frac{\alpha(1-\alpha)}{NTOT}}$$

where $NTOT$ is the number of replications used in our Monte Carlo experiments.

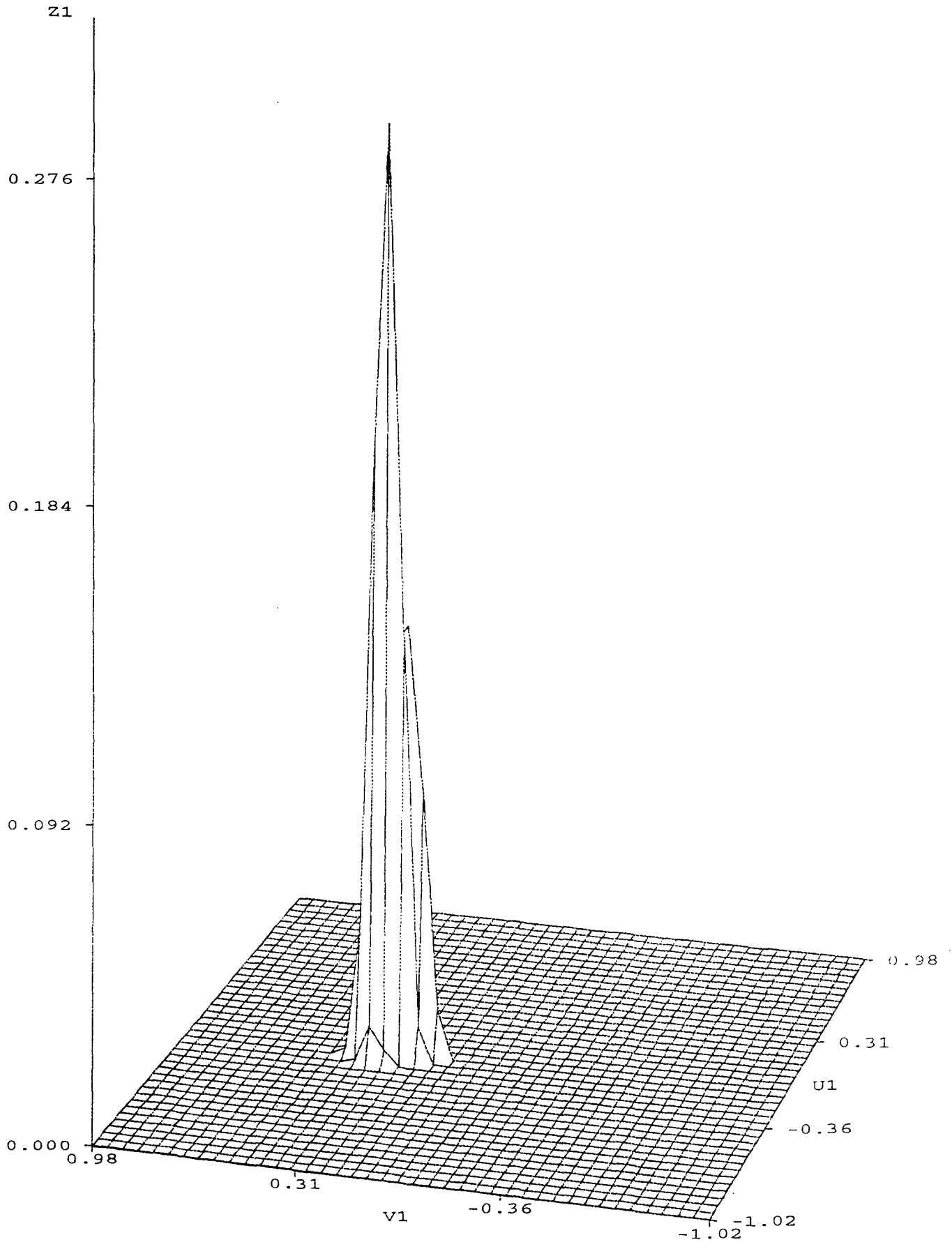
Secondly, we compare the two GOF test results. That is, we look at the P_r results from the modified Q_n test. And we compare them to the P_r results from the Exact GOF tests for the



Generating from SIRP x=s*z, p=2, rho=0.5.

Figure 1.

U_n vs V_n plot, for n=50 and 5000 replications.



Generating from SIRP x=s*z, p=2, rho=0.5.

Figure 2.

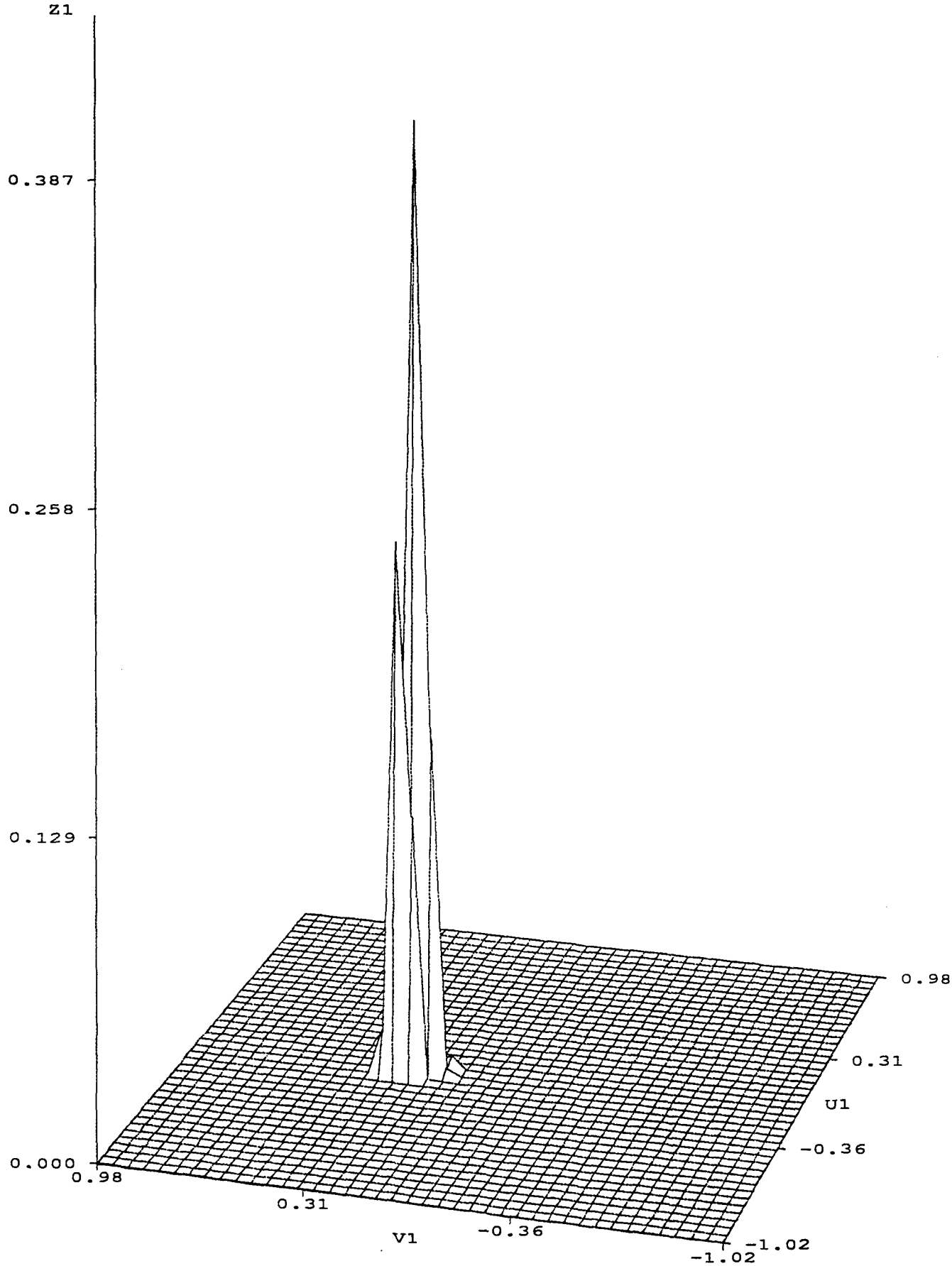

U_n vs V_n plot, for n=200 and 5000 replications.
Z1
0.387
0.258
0.129
0.000
0.98
0.31
-0.36
-1.02
V1
0.98
0.31
-0.36
-1.02
U1
Generating from SIRP x=s*z, p=2, rho=0.5.

Figure 3.

Rayleigh distribution (of the transformation w of p_N). The Exact Rayleigh distribution is tested using a standard GOF procedure (say the Chi Square or Kolmogorov Smirnov).

In Table 3 we show a comparison of these percent rejections (P_r) obtained by Monte Carlo from both of these GOF tests. They correspond to the case $N = 2$ and $\rho = 0.5$ (Bivariate K-Distributed SIRP X). For this special case we can use the Exact GOF test because we **know** the theoretical distribution of the transformation w, i.e. of the quadratic form p_N.

Table 3. Results for Exact vs. Modified Q_n GOF Tests ($N = 2$)

GOF Test	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$	n
Exact	0.0756	0.0386	0.0064	10
Q_n	0.0848	0.0398	0.0098	10
Exact	0.0838	0.0354	0.0068	25
Q_n	0.0930	0.0410	0.0092	25
Exact	0.0936	0.0468	0.0098	50
Q_n	0.0912	0.0426	0.0090	50
Exact	0.0956	0.0450	0.0090	100
Q_n	0.0874	0.0444	0.0124	100
Exact	0.0970	0.0520	0.0080	200
Q_n	0.1000	0.0610	0.0170	200

Results in Table 3 were obtained by simulating NTOT samples from a Bivariate K SIRP. Then, by testing the GOF of the corresponding transformation w of the quadratic function p_N, via a standard Chi Square GOF test (Exact). And again, by testing the GOF of the same simulated sample, now directly on p_N, using with our modified Q_n. Throughout this experiment we assume that the covariance matrix Σ is known and we use it to obtain p_N. The experiment was conducted for $NTOT = 5000$ replications and for a bivariate correlation $\rho = 0.5$, with sample sizes $n = 10, 25, 50, 100, 200$.

Notice how (i) the values of the Exact GOF test for p_N and (ii) the modified Q_n test (when Σ is known) are relatively close. Notice also that, as expected, both tests converge to their nominal significance level (α) as the sample size (n) increases. But the modified Q_n converges at a slower rate.

In Table 4 we present the percent rejection results for $N = 8$. This time we **do not** have the Exact distribution of the quadratic form p_N, to compare with, as above. However, we can compare the percent rejections of the modified Q_n test with the nominal significance levels ($\alpha = 0.1, 0.05, 0.01$). Observe how our proposed Q_n test remains somewhat conservative (e.g. for $\alpha = 0.1, 0.05, 0.01$ we get, approximately, $P_r = 0.085, 0.045, 0.011$). However, the test still slowly moves toward the nominal α.

Table 4. Modified Q_n GOF Test Results For N=8.

GOF Test	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$	n
Q_n	0.08980	0.04200	0.00720	25
Q_n	0.08648	0.04400	0.00972	50
Q_n	0.08476	0.04504	0.01160	100
Q_n	0.08220	0.04160	0.01190	200

We investigated further how close these two GOF tests (Exact (E) and modified Q_n) really are. We took their percent rejection difference. That is, for every pair (E, Q_n) with the same setting $(n, 2, \rho)$, we obtained the difference percent rejections:

$$\delta_1 = P_r(E) - P_r(Q_n)$$

The descriptive statistics corresponding to these δ_1 values are presented in Table 5. Notice that this analysis is performed for $N = 2$ only and includes correlations $\rho = 0.0, 0.5, 0.9$.

Table 5. Descriptive Statistics for $\delta_1 = P_r(E) - P_r(Q_n)$.

Statistic	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Mean	0.0039	0.0013	-0.0023
Median	0.0098	0.0020	-0.0020
Std-Dev.	0.0132	0.0057	0.0013
Min	-0.0162	-0.0096	-0.0040
Max	0.0171	0.0080	0.0002
Q_1	-0.0100	-0.0025	-0.0035
Q_3	0.0161	0.0069	-0.0014
L_B	-0.0045	-0.0023	-0.0031
U_B	0.0123	0.0049	-0.0015

Notice how the mean/median of these differences (δ_1) are relatively small. And how the fiducial intervals corresponding to the Range (Min, Max), Interquartile Range (Q_1, Q_3), and the standard 95 percent t-interval (L_B, U_B), cover zero. This means that Zero can be an acceptable value for this difference (i.e. the two tests may be statistically equivalent).

However, a graphical analysis of these data, on the settings (n, N, ρ), shows how the differences δ_1 increase in sample size (n) from negative to positive values, then stabilize. This confirms the existence of a (small) **Bias** (e.g stable difference) between the two, even for large n.

A regression analysis of $\delta_1 = f(n, \rho)$ confirms that (i) sample size is a significant factor but that (ii) intercorrelation ρ is not. And that (iii) the effect of sample size (n) decreases as the nominal significance level (α) decreases. The latter means that, for small values of (α) we do not improve

much on our modified Q_n by taking a larger sample. That is, the Bias remains when testing at $\alpha = 0.01$.

The Bias detected (e.g. δ_1^*), for a large sample size (n) was estimated as $\delta_1* = 0.015, 0.006, 0.002$ for significance levels $\alpha = 0.1, 0.05, 0.01$. This is about 15 percent below the nominal level.

We thus conclude that our proposed Q_n GOF test is adequate for testing the GOF of a multivariate SIRP. We recognize that Q_n is somewhat conservative with respect to the Exact test. But considering that, in most situations, we lack any other statistic to test the GOF of a multivariate SIRP X, we proceed to use it.

PHASE III: EVALUATION OF SIRP FACTORS AND ESTIMATORS Σ^*.

The Q_n GOF test can be applied in two different ways. First, as a (conservative) GOF test for (identifying) SIRP's. And second, as a research tool to assess different characteristics of the SIRP sample (e.g. the effect of n, N, ρ). This second application is the objective of the next phase of our research.

In the previous two phases we successfully constructed and validated the modified Q_n GOF test, for the identification of a multivariate SIRP X. The validation was implemented using the special case of the Bivariate K SIRP. As **Standard** we used the **Exact distribution** of the transformation (w) of the quadratic form p of this special case. We also compared more general K-Distributed SIRP's (for $N = 4, 8$) and verified how the modified Q_n test still works acceptably. We can now rely on our proposed Monte Carlo derived test Q_n to identify a general SIRP X, via its associated quadratic form p_N.

But, in addition, we can use the Q_n GOF test developed above, as a research tool. It is in this capacity that Q_n can help to (i) *assess the performance of a proposed estimator* of p_N and to (ii) *analyze the effects of sampling characteristics* (n, N, ρ) of the SIRP X. This is the objective of the present and last phase of this research.

To this effect, we perform another Monte Carlo experiment. We now simulate $NTOT = 25000$ samples from the same SIRP X as in Phases I and II. We use vector sizes $N = 2, 4, 8$, sample sizes $n = 25, 50, 100, 200$ and intercorrelations $\rho = 0.0, 0.5, 0.9$. In this experiment we obtain two quadratic forms: $p_N = X'\Sigma^{-1}X$ and $p_N^* = X'\Sigma^{*-1}X$. We obtain them by first using the known covariance matrix (Σ). Then, we obtain them from the simulated sample (Σ^*), using the maximum likelihood estimator of Σ. We then apply the modified GOF test Q_n to both resulting quadratic forms, p_N and p_N^*. And we use the GOF test results to assess the estimator Σ^*.

A graphical representation of the Experimental Design described above for our three factor (n, N, ρ) Monte Carlo experiment, is shown in Figure 4. The percent rejection results obtained from such experiment are shown in Table 6.

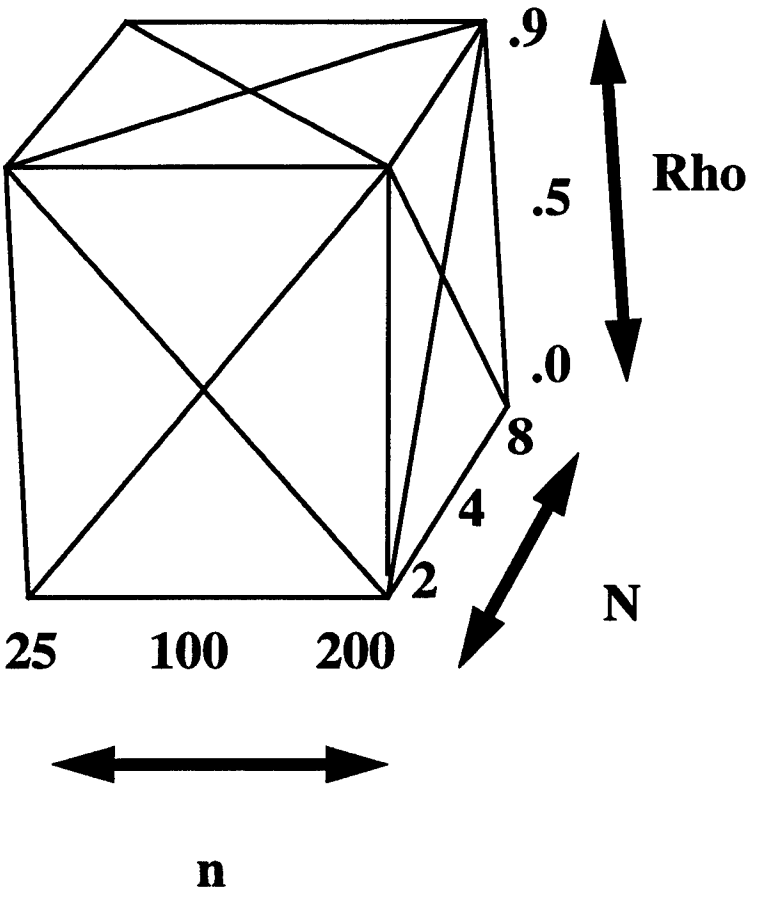


Figure 4: Representation of the Design

Table 6. Results for the General Factorial Experimental Design.

Test	n	N	ρ	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Exact	25	2	0.0	0.0826	0.0402	0.0064
Q_n	25	2	0.0	0.0930	0.0430	0.0094
Exact	25	2	0.5	0.0774	0.0352	0.0050
Q_n	25	2	0.5	0.0936	0.0448	0.0090
Exact	25	2	0.9	0.0870	0.0414	0.0062
Q_n	25	2	0.9	0.0936	0.0429	0.0102
Exact	50	2	0.5	0.0949	0.0465	0.0094
Q_n	50	2	0.5	0.0880	0.0450	0.0092
Exact	100	2	0.0	0.0978	0.0466	0.0098
Q_n	100	2	0.0	0.0850	0.0440	0.0116
Exact	100	2	0.5	0.0983	0.0489	0.0096
Q_n	100	2	0.5	0.0824	0.0409	0.0109
Exact	100	2	0.9	0.0968	0.0494	0.0095
Q_n	100	2	0.9	0.0828	0.0416	0.0115
Exact	200	2	0.0	0.0984	0.0460	0.0090
Q_n	200	2	0.0	0.0839	0.0442	0.0123
Exact	200	2	0.5	0.0970	0.0482	0.0096
Q_n	200	2	0.5	0.0808	0.0412	0.0109
Exact	200	2	0.9	0.1006	0.0508	0.0103
Q_n	200	2	0.9	0.0813	0.0423	0.0125
Q_n	25	4	0.0	0.0918	0.0426	0.0075
Q_n	25	4	0.5	0.1002	0.0453	0.0081
Q_n	25	4	0.9	0.0894	0.0429	0.0079
Q_n	50	4	0.5	0.0942	0.0477	0.0106
Q_n	100	4	0.0	0.0917	0.0465	0.0129
Q_n	100	4	0.5	0.0872	0.0454	0.0113
Q_n	100	4	0.9	0.0879	0.0453	0.0109
Q_n	200	4	0.5	0.0832	0.0432	0.0133
Q_n	25	8	0.0	0.0893	0.0430	0.0078
Q_n	25	8	0.5	0.0928	0.0439	0.0074
Q_n	25	8	0.9	0.0898	0.0438	0.0085
Q_n	50	8	0.5	0.0842	0.0426	0.0096
Q_n	100	8	0.0	0.0808	0.0408	0.0106
Q_n	100	8	0.5	0.0890	0.0442	0.0112
Q_n	100	8	0.9	0.0854	0.0434	0.0113
Q_n	200	8	0.5	0.0822	0.0416	0.0119

A first look at Table 6 confirms that the modified Q_n test, for $N = 2, 4, 8$ and various ρ's and sample sizes (n), remains stable at a (conservative) small distance of the Exact test. That is, Q_n is adequate. However, we now want to investigate the behavior of this modified Q_n test when the covariance matrix (Σ) of the SIRP process X is not known. We want to assess what happens to the modified Q_n when Σ is estimated from the data.

To this effect, we concentrated our analyses in studying the variations of the modified Q_n test results. In Table 7 we show the descriptive statistics for the percent rejections (P_r) of the modified Q_n GOF test, for the General Experiment of Table 6. Notice again the fiducial intervals (Min, Max), (Q_1, Q_3) and (L_B, U_B). They still show a lag of about 15 percent between the percent rejections of the modified Q_n test and the nominal α's (i.e Q_n is a conservative test).

Table 7. Descriptive Statistics for The Q_n Data in Table 6.

Statistic	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Mean	0.0884	0.0439	0.0104
Median	0.0880	0.0435	0.0106
Std-Dev.	0.0054	0.0034	0.0020
Min	0.0808	0.0398	0.0072
Max	0.1002	0.0610	0.0170
Q_1	0.0841	0.0422	0.0090
Q_3	0.0923	0.0450	0.0116
L_B	0.0866	0.0428	0.0095
U_B	0.0902	0.0450	0.0111

We proceeded to investigate the effects of estimating the covariance matrix (Σ^*) in the modified Q_n test. In Table 8 we show the percent rejections (P_r) for the experiment implemented to compare these two statistics: p_N, p_N^*.

For this experiment, we ran a series of settings (n, N, ρ) for the values in Table 8. We fixed (a representative) intercorrelation $\rho = 0.5$. We implemented the modified Q_n through the quadratic form p_N, i.e. using the known covariance ($Q_n(\Sigma)$). And we implemented Q_n through the quadratic form p_N^*, i.e. estimating the covariance matrix from the sample ($Q_n(\Sigma^*)$). In Table 8 these two cases are denoted, respectively, **Known** and **Estim.**

We analyzed the pooled data using multiple regression, i.e. $P_r = f(n, N, p_N, p_N^*)$. And we reconfirmed the results obtained above, i.e. that (i) sample size is a significant factor and that (ii) intercorrelation ρ is not a factor in the performance of the modified Q_n GOF test.

We then calculated the differences in percent rejections between the two versions of the Q_n GOF test (with and without knowledge of Σ):

$$\delta_2 = P_r(\Sigma) - P_r(\Sigma^*)$$

Results of the corresponding descriptive statistics of δ_2 are shown in Table 9.

Notice in Table 8 that, as expected, sample size (n) is indeed a significant factor. As the sample size increases, the performance of the modified Q_n test obtained with the estimated covariance matrix ($Q_n(\Sigma^*)$) gets closer to the one obtained using the known covariance matrix ($Q_n(\Sigma)$).

We investigated this situation further, both graphically and through regression analysis. We used, as dependent variable, the differences δ_2 defined above. Both of these analyses confirmed our initial observations. Inter variate correlation (ρ) is not significant and vector size (N) is significant to the extent that it determines the **adequate sample size** required to safely implement the modified Q_n GOF test. These adequate sample size values were found to be:

(1) For $N = 2$ we need $n > 25$

(2) For $N = 4$ we need $n > 100$

(3) For $N = 8$ we need $n > 400$

Table 8. Effect of Covariance Matrix Estimation in P_r.

Σ	n	N	ρ	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Known	10	2	0.5	0.08480	0.03980	0.00980
Estim	10	2	0.5	0.14420	0.07840	0.02060
Known	25	2	0.5	0.09300	0.04100	0.00920
Estim	25	2	0.5	0.10020	0.05780	0.01500
Known	50	2	0.5	0.09120	0.04260	0.00900
Estim	50	2	0.5	0.08300	0.04120	0.01180
Known	100	2	0.5	0.08740	0.04440	0.01240
Estim	100	2	0.5	0.07660	0.03620	0.00820
Known	200	2	0.5	0.10000	0.06100	0.01700
Estim	200	2	0.5	0.09000	0.04100	0.01200
Known	25	4	0.5	0.10020	0.04530	0.00810
Estim	25	4	0.5	0.22920	0.14430	0.05130
Known	100	4	0.5	0.08720	0.04540	0.01136
Estim	100	4	0.5	0.10380	0.05876	0.01724
Known	25	8	0.5	0.08980	0.04200	0.00720
Estim	25	8	0.5	0.94400	0.86200	0.55580
Known	50	8	0.5	0.08648	0.04400	0.00972
Estim	50	8	0.5	0.52872	0.38176	0.17040
Known	100	8	0.5	0.08476	0.04504	0.01160
Estim	100	8	0.5	0.28084	0.18348	0.06648
Known	200	8	0.5	0.08220	0.04160	0.01190
Estim	200	8	0.5	0.16090	0.09290	0.02830

Regression and graphical analyses of $\delta_2 = f(n, N, \rho)$, for $\alpha = 0.1$, indicate that the differences between $Q_n(\Sigma)$ and $Q_n(\Sigma^*)$ dissapear, as (n) increases and N decreases (i.e. the corresponding

regression independent terms are not significant). And this significant effect of (n, N), tends to decrease as the GOF test significance levels $\alpha = 0.05, 0.01$ (i.e. GOF results with and without knowledge of the covariance matrix tend to agree, when testing at these small values of α).

Table 9. Descriptive Statistics for $\delta_2 = P_r(\Sigma) - P_r(\Sigma^*)$.

Statistic	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Mean	-0.1595	-0.1351	-0.0763
Median	-0.0594	-0.0386	-0.0108
Std-Dev	0.2666	0.2490	0.1637
Max	0.0108	0.0200	0.0050
Min	-0.8542	-0.8200	-0.5486
Q_1	-0.1961	-0.1384	-0.0549
Q_3	0.0082	0.0014	-0.0028
L_B	-0.3387	-0.3024	-0.1864
U_B	0.0197	0.0322	0.0337

We also performed regression analyses for the subsets of tests done when Σ was known and when Σ was estimated from the data. When Σ was known (e.g. $Q_n(\Sigma)$) regressions showed a mild effect of n but no effect of N. And these factor effects became stronger as test level α decreased.

On the other hand, regression analyses for the case $Q_n(\Sigma^*)$, when Σ was estimated from the sample, showed as expected a significant effect of both n and N. This confirms our previous results about the estimation of the covariance matrix. They also reinforce our concern about establishing adequate values for n, depending on the values of N.

Summarizing the results of our comparison of the two versions of the modified Q_n GOF test, i.e. when the covariance matrix Σ is known and when it is estimated from the data, we have:

(1) There is no effect of intercorrelation ρ. This is the strongest result of this research. For, in most cases, the covariance matrix Σ is unknown. And if every time it is estimated from the sample there would be a problem with the (unknown) underlying correlation, this estimation would be very problematic.

(2) When estimating Σ from the sample, a larger sample size provides a better statistic p_N^* and hence, a better GOF test Q_n. This is not as necessary when Σ is known.

(3) The selection of an **adequate** sample size (n) depends on the vector size (N) being used. This is especially important when estimating the covariance matrix Σ from the sample. For the adequate values, see above.

(4) With adequate sample and vector sizes the test $Q_n(\Sigma^*)$, i.e. the one performed by estimating the covariance matrix, is approximately as good as $Q_n(\Sigma)$, the one performed using the true covariance matrix.

FUTURE RESEARCH:

The GOF testing methodology presented in this document has application in identification procedures of SIRP radar studies. We can test any SIRP via its quadratic form p_N, by using the modified Q_n. We can implement any sample size n, vector size N, covariance structures Σ and other SIRP processes X, with this methodology, in exactly the same way. And, given the rapidly expanding computing power, the problems of implementing a computationally intensive method like the one developed here, are no longer valid.

In particular, this methodology has application in conjunction with the **distribution approximation methodology of Kaman (1992)**. There, approximation charts were provided. And an incoming signal (SIRP X) was identified by plotting Q_n into the approximation chart. The selected distribution would be, in the approximation chart, the one closest to the plotted point.

In many cases, a plotted point may lie close to two or more candidate distributions. And we would be forced to choose on a subjective or ad-hoc basis. With a GOF test criteria, like the one implemented in this research, the selection between several candidate distributions could be performed on a probabilistic basis. We would select that distribution with the largest p-value, from all the **modified** Q_n GOF tests performed on all approximation chart candidates.

Another possible use of the modified Q_n consists in pursuing further the investigation of factors of interest (e.g. n, N, ρ) with additional SIRP families (other than K-Distributed or the same but with different parameters). Or in investigating other methods of estimation of the covariance matrix Σ of the SIRP process. The procedure would be similar to the one described in Phase III of this report.

Finally, it may be possible to map a physical clutter scenario (e.g. a forest, hills) into a specific SIRP function. And it may be possible to determine its parameters. In this case, a Monte Carlo study may yield the parameters of the modified Q_n test required to implement the GOF test of any incoming signal, for different settings (n, N, ρ).

By building a library with such parameters (e.g. a Research and Classification Parameter Bank) it may be possible to quickly identify an incoming signal as belonging to one of such scenarios. And, with more time and resources, we may obtain more accurate parameter values, closing the small (conservative) Bias of the Q_n test. For example, a larger Monte Carlo experiment, performed in a faster computer, with higher precision, might yield better estimators. And the resulting modified Q_n GOF test could be more robust and accurate than the present version.

CONCLUSIONS.

This research report concludes the validation effort started in Romeu (1992a). We have shown that the theoretical SIRP radar clutter modelling procedure given in Kaman (1992) is computer implementable and fulfills its intended purpose of (*i*) generating SIRP's and (*ii*) identifying an incoming signal. We have done the latter by showing that the quadratic form p_N is a good statistic for testing the GOF of any SIRP. In Romeu (1992a) this procedure was implemented for two special cases: the Gaussian and the unidimensional K-Distributed SIRP. In the present research it was completed for the special case of the two-dimensional K-Distributed SIRP and for the general ($N > 2$) K-Distributed SIRP.

In addition, we have demonstrated a GOF testing methodology for identifying a General SIRP: the modified Q_n GOF test. This Monte Carlo based testing methodology was also validated. We also showed how the Q_n GOF test is an adequate (though) conservative general procedure for identifying a multivariate SIRP.

We have investigated the maximun likelihood estimator of the covariance matrix Σ of an SIRP process X. We have shown how, if the sample size is adequate for the vector size of X, the estimation is good and the corresponding modified Q_N GOF test provides good results. We give, in Phase III of this report, ballpark values for the adequacy of the (n, N) sizes required.

We have investigated the effects of three factors that can affect the modified Q_n GOF test. These are (n, N, ρ). We have shown how the intercorrelation (ρ) is not a significant factor. The other two, can affect the test in the sense that the (sample and vector) sizes may not be adequate for its correct implementation.

Finally, some other potential uses of the modified Q_n GOF test have been discussed and a roadmap for a possible extension of this research has been proposed.

ACKNOWLEDGEMENTS.

The present research has benefited from the cooperation and collaboration of Drs. Don Weiner (Syracuse University), Jim Michels (Rome Labs) and Murali Rangaswamy (Rome Labs; Hanscomb, AFB). We particularly thank our Rome Lab Focal Point, Dr. Michels, for his constant support. We appreciate the computer and logistic support of the Cortland College of the State University of New York and of the CASE Center of Syracuse University.

REFERENCES

ATT, *S-Plus Statistical Software: User's Manual*, 1992.

Bratley, P., Fox, B. and L. Schrage, *A Guide to Simulation publ Springer-Verlag*, 1983.

Chambers, J. and T. Hastie, *Statistical Models in S*, Wadsworth, 1992.

Cochran, W. and G. Cox, *Experimental Design*, Wiley, 1992.

David, H., *Order Statistics*, Wiley, 1981.

Dudewicz, E. and T. Ralley, *The Handbook of Random Number Generation and Testing with TESTRAND Computer Code*, American Sciences Press, Inc., 1981.

Johnson, M. E., *Multivariate Statistical Simulation*, Wiley, 1987.

Johnson, N. and S. Kotz, *Continous Univariate Distributions (Vols. 1 and 2)*, Wiley, 1970.

Kaman Sciences, *Contract Report No. F36602-89-C-0082, Task 10*, Rome Lab Technical Report, 1992.

Kennedy, W. and J. Gentle, *Statistical Computing*, Wiley, 1990.

Kendall, M. G. and A. Stuart, *The Advanced Theory of Statistics (Vols. I, II and III)*, Charles Griffin and Co., London, 1966.

Ozturk, A. and E. Dudewicz, *A New Statistical Goodness of Fit Test Based on Graphical Representation*, Bimetrical Journal **34(4)** (1992), 403–427.

Ozturk, A. and J. L. Romeu, *A New Graphical Test for Multivariate Normality*, Comm. in Statist. (Simula.) **21(1)** (1992).

Rangaswamy, M., *Spherically Invariant Random Processes for Radar Clutter Modeling, Simulation and Distribution Identification*, Ph.D. Dissertation. Syracuse University. December 1992..

Rangaswamy, M.; Weiner, D. and A. Ozturk, *Computer Generation of Correlated Non Gaussian Clutter for Radar Signal Detection*, IEEE Trans. Aerosp. Electr. Sys. (1992).

Rangaswamy, M.; Weiner, D. and A. Ozturk, *Simulation of Correlated Non Gaussian Interference for Radar Signal Detection*, Proceedings of the 25th Annual ASILOMAR Conference on Signals, Systems and Computers (1991).

Romeu, J. L., *Teaching Statistics With Simulation: A classroom experience*, Journal of the Institute of Statisticians **35(4)** (1986).

Romeu, J. L., *Monte Carlo Validation of a Theoretical Model for the Generation of Non Gaussian Radar Clutter.*, AFOSR Summer Research Report Compendium, 1992a.

Romeu, J. L., *Further Discussion on Monte Carlo Validation of a Theoretical Model for the Generation of Non Gaussian Radar Clutter*, CASE Center Technical Report No. 9209. Syracuse University, 1992b.

Romeu, J. L., *Further Monte Carlo Studies of a Theoretical Model for Non Gaussian Radar Clutter Characterization*, RIP Research Proposal. AFOSR, (1992c).

Romeu, J. L., *Validation of Multivariate Monte Carlo Studies*, Proceedings of the IMSIBAC-4, San Sebastian. Spain (1992d).

Romeu, J. L., *Monte Carlo Investigation of a Model for Generation of Non Gaussian Radar Clutter*, Proceedings of the 25th Anniversary of the Interface Symposium (1993a).

Romeu, J. L., *Monte Carlo Study of a Non Gaussian Radar Clutter Generator*, Rome Lab Technical Report (in preparation) (1993b).

Romeu, J. L. and A. Ozturk, *A Comparative Study of GOF Tests for Multivariate Normality*, Journal of Multivariate Analysis **46** (1993c).

Rubinstein, R., *Simulation and the Monte Carlo Method*, Wiley, 1981.

Shapiro, S. and A. Gross, *Statistical Modeling Techniques*, Marcel Dekker, 1981.

Widman, L., K. Loparo and N. Nielsen, *Artificial Intelligence, Simulation and Modeling*, Wiley, 1989.

HIERARCHICAL MODELING AND SIMULATION

Robert G. Sargent
Professor
and
Douglas G. Fritz
Graduate Student

Simulation Research Group
College of Engineering and Computer Science
Syracuse University
Syracuse, NY 13244

Final Report for:
Summer Research Extension Program

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, DC
and
Syracuse University

December 1993

HIERARCHICAL MODELING AND SIMULATION

Robert G. Sargent
Professor
and
Douglas G. Fritz
Graduate Student

Simulation Research Group
College of Engineering and Computer Science
Syracuse University

Abstract

A general paradigm using two complementary types of hierarchical model specification structures is developed for the specification of discrete event simulation models based on encapsulated atomic model components that communicate solely via message passing. One type of structure specifies the hierarchical coupling of model components and how the atomic components of the model can interact. The other type of structure specifies the internal behavior of each atomic component in a hierarchical way. The model elements of both specifications are encapsulated and reusable. Hierarchical Control Flow Graph Models, a special case of this general paradigm, is developed as a hierarchical extension of Cota and Sargent's Control Flow Graph Model representation. Algorithms are presented for transforming Hierarchical Control Flow Graph Models into equivalent nonhierarchical Control Flow Graph Models which can then be executed using existing algorithms for Control Flow Graph Models.

HIERARCHICAL MODELING AND SIMULATION

Robert G. Sargent
Douglas G. Fritz

1 Introduction

We are interested in a type of simulation called Discrete Event Simulation (DES). Discrete event simulation is a simulation where instantaneous state changes (events) occur only at arbitrarily (usually random) spaced points in time, and no state changes occur between the events. Discrete event simulation is used to model and analyze a large range of systems which include queueing network systems, computer and communications systems, military systems, flexible manufacturing systems, and traffic flow systems. Some simulation languages which support DES are GPSS, RESQ, SIMULA, SLAM, SIMAN, SIMSCRIPT, and SIM++.

There are different *world views*, or ways to view a system in order to model it [Bal88]. The process interaction world view is a world view in which the modeler models the system as a set of interacting processes where each process generally represents some component of the system under study. This world view is a relatively natural way to view and model a system.

In the classical process interaction world view, processes interact by reading and modifying the internals of other processes (e.g., scheduling or canceling events for each other). There are disadvantages to this type of interaction in which processes require knowledge of the internals of other processes. One method of addressing this issue is to encapsulate the processes and require all process interaction to use a strictly defined interface (see Cota and Sargent [CS92]). Encapsulated processes can communicate with each other via methods such as shared variables or message passing. If the only method of interaction between processes is message passing, then by default, the processes are encapsulated. In this paper we are concerned only with the message passing method.

In this "modified" process interaction world view a model is composed of a set of encapsulated processes which communicate solely via message passing. Defining a model using this world view requires two complementary types of specifications. One type (the coupling specification) specifies a list of processes in the model and the interprocess coupling which defines how these processes can interact (i.e, the routing (connectivity) and transformation of interprocess messages). The second type of specification (behavior) is used to specify the causal behavior of each of the processes in the model. In DES we must understand the behavior of the system components well enough to model the causal relationships which define the behavior of the individual processes. Thus far we have been using the term "process" to refer to a model of a system component. We also use "model component" as a synonym for "model process" and when the usage is unambiguous we sometimes drop the "model" part and use simply "component". In general, a complete model specification in this modified world view has a single coupling specification which specifies a list of the model components in the

model and their intercomponent coupling and a set of behavior specifications (one per model component) which specify the causal relationships which generate the behavior of the individual model components.

Cota and Sargent developed a model representation called Control Flow Graph (CFG) Models [CS90a] based on the modified process interaction world view in which encapsulated model components interact via message passing. The primary objective of CFG Models was to make information useful for parallel simulation explicit in the model representation. Cota and Sargent then developed a set of algorithms [CS90c] for the execution of CFG Models that allow CFG Models to be executed on either sequential architectures or parallel/distributed architectures without any additional modeler input. The parallel algorithms generate "automatic lookahead" information which is a key element in the efficiency of parallel simulation [Fuj90]. The ability to use these existing algorithms for model execution is a desirable property of this model representation.

In a CFG Model, the model components and their interconnections are specified via an Interconnection Graph which specifies how messages are routed between components. Intercomponent messages flow over *channels* (unidirectional message paths) which connect one model component to another model component. Each type of message is normally sent over a different channel, so there can be multiple channels from one component to another component with each channel carrying a different type of message. As the number of model components in a model and the number of message types in a model grow, the Interconnection Graph can become complex.

The causal behavior of each model component in a CFG Model is specified via a CFG where the nodes in the CFG specify "control states" of the model component and the edges specify possible component control state transitions. We use "control state" in this discussion as it is defined in [CS92] where a "control state is the formalization of a process reactivation point" or as an element of the sequential state set of a component as in DEVS [Zei84a]. Associated with each edge are three attributes: a *condition*, a *priority*, and an *event* (state transition function). The condition specifies when an edge can become a candidate for traversal, the priority is used to break ties when more then one edge is a candidate for selection, and the event specifies a state transition for the model component if that edge is selected. Conceptually, each model component has a "current" control state or "point of control". The model component examines the condition attributes (and the priority attributes, if necessary) on all edges leaving the current control state and selects one of these edges for traversal. The point of control then traverses the selected edge, executing the event associated with that edge and setting the new "current" control state to the control state that the selected edge is incident on. (Another way to view this is that the point of control moves from the current control state across the selected edge to the next control state.) This operation is repeated until the simulation terminates. One concern with using CFG's to specify the causal behavior of components is that the CFG representation may become difficult to generate as the complexity of the behavior of the component modeled increases.

Modeling is an intellectually challenging activity, particularly when the system under consideration becomes complex, and humans have a limited capacity for dealing with complexity. The "magic number" of 7 ± 2 represents the number of pieces of information that the typical person can keep in mind at a time.

Thus, to aid in the modeling of complex systems, it is desirable that the modeler be concerned only with on the order of seven pieces of information at any one time.

Booch[Boo91] specifies three common principals for dealing with complexity: decomposition, abstraction, and hierarchy. Decomposition is the divide and conquer principle where the system is divided into parts and each part is then addressed in turn. Abstraction allows the modeler to work with fewer pieces of information at a time by concentrating only on the aspects of the system entity that are relevant to the current objectives while effectively ignoring those aspects that have little or no impact on the current system. Hierarchy applies the divide and conquer strategy of decomposition in a recursive manner which allows the modeler to recursively decompose components into subcomponents until such a point is reached where each of the remaining components is simple enough to comprehend and model as a whole without further decomposition. Thus, hierarchical modeling allows us to model complex systems easier than is possible without hierarchical modeling.

In this paper we first develop a general paradigm for the specification of models based on encapsulated model components which interact via message passing (Cota and Sargent's "modified" process interaction world view [CS92] and message passing). We describe structures for each of two complementary types of specifications (coupling and behavior) which provide support for hierarchical model specification. Since the coupling specification is sufficiently narrow (message passing only), we also develop a systems theoretic formalism for the coupling specification structure.

We then develop two hierarchical extensions (one for each type of specification) to CFG Models to obtain "Hierarchical Control Flow Graph (HCFG) Models". These extensions are special cases of our general paradigm. One extension, the Hierarchical Interconnection Graph, extends the Interconnection Graph notion by supporting hierarchical coupling specification using "coupled" components. This allows the modeller to couple together existing model components to form new model components. The other extension to CFG Models, Hierarchical Control Flow Graphs (HCFG's), is an extension of the CFG concept. In this extension the modeler can use "Macro Control States" (MCS's) in addition to control states when modeling a component's behavior. A macro control state encapsulates one or more control states and/or macro control states and their associated edges (note the recursive definition). Edges originating at encapsulated control states have the set of attributes (condition, priority, and event). (Macro control states encapsulate a partial behavior of a model component.) Support for recursive decomposition of the behavior specification provides the modeler with a hierarchical structure which can aid in the specification of larger and more complex model components.

The organization of the remainder of this paper is as follows. In section 2 we discuss the modified process interaction world view and why encapsulation of components is desirable. In section 3 we discuss the two types of specifications required for a model specified using encapsulated components and message passing. Section 4 presents a brief overview of the CFG Model representation. Hierarchical Control Flow Graph Models are developed in Section 5. In section 6 we present algorithms for mapping an HCFG Model into

a CFG Model. Section 7 summarizes this paper. (A longer version of this paper which contains additional sections and subsections is [FS93].)

2 Modularity of Components

Cota and Sargent [CS92] specify a modification of the process interaction world view in which components have the modularity property. The modularity property refers to a combination of the locality and encapsulation properties.

We use the definition of locality as used in [CS92] which can be applied to models in addition to programs. In [Wei71], locality is defined as "that property when all relevant parts of a program are found in the same place". We use the definition of encapsulation as defined in [CS92]. "Encapsulation of a component in a simulation model refers to the complete definition of the state set of that component and the behavior of that component independently of the state set or behavior of any other component. A component is encapsulated if the state set and behavior of that component can be changed (by a modeler) without changing the state set or behavior of any other component in the model."

Model components in the classical process interaction world view do not possess the modularity property. Components typically interact by reading and/or modifying the internals of other components. This type of interaction where one component reads and modifies the internals of another component means that model components must be designed using knowledge of the internals of the components that it interacts with. This interdependency of model components can be an impediment to desirable capabilities which include hierarchical modeling, model component reuse, and distributed simulation execution.

Cota and Sargent[CS92] address these difficulties with a modification of the process interaction world view in which they apply the concept of modularity to the model components. Modularity of model components combines the properties of locality of information in a component with the encapsulation of information within that component. The information needed to simulate a model component is contained entirely within that component and all access to the "outside world" is via a strictly defined interface. This corresponds roughly to the software engineering concept of separation of interface and implementation. (See Cota and Sargent[CS92] for advantages of modularity of components.)

When we encapsulate model components, the components still need to communicate relevant information with each other. Encapsulated processes can communicate with each other via methods such as shared variables or message passing. We are concerned in this paper with only the message passing method of intercomponent communication.

Message passing can use either a passive receiver or an active receiver protocol [CS92]. In passive receiver message passing, (as generally used in object-oriented DES such as Zeigler's DEVS-Scheme[Zei90] and Sim++ [GB89]) messages are received and handled by the receiving model component as they arrive. This contrasts to active receiver message passing in which incoming messages are queued at the receiving

component to be handled at a (potentially later) time as chosen by the receiving component.

During simulation execution, a component may be in a state where the arrival of certain types of messages will not affect that component's behavior. In these cases active receiver message passing gives a component the option of temporarily ignoring these messages. Simulation languages and representations which support this modification of the process interaction world view as specified by Cota and Sargent[CS92] include May [BCM87], Maisie [BL90], and Control Flow Graph Models[CS90a].

3 Modeling and Model Specification

In this section we we develop a paradigm for the specification of models based on encapsulated model components that interact via message passing. This includes object-oriented model representations such as Zeigler's DEVS-Scheme [Zei90] and Sim++[GB89] as well as model representations based on Cota and Sargent's [CS92] modified process interaction world view. A complete model specification uses two complementary types of specification structures, a *coupling* specification and a *behavior* specification. Some model representations support hierarchical mechanisms for one or both types of specification structures. We describe structures for each of the two specification types which provide support for hierarchical model specification. Since the coupling specification is sufficiently narrow (message passing only), we also develop a systems theoretic formalism for the coupling specification structure.

We can look at a complete model specification as consisting of two parts as shown in Figure 1: an "above the table" coupling part which specifies the model structure outside the encapsulation boundaries of the model components and a "below the table" behavior part where the behavior structures for the internal model components are specified. We first look at the behavior specification of a component and then the coupling of components.

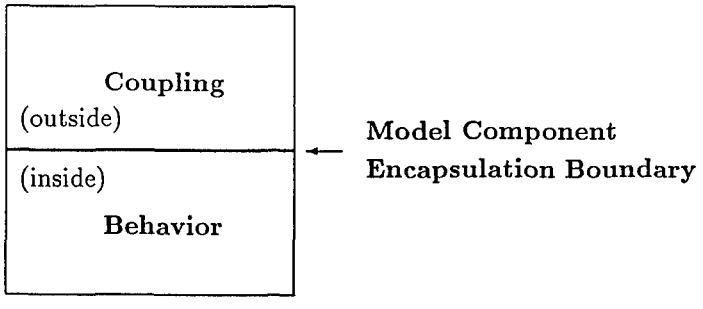


Figure 1: Outside versus Inside of a Model Component

3.1 Atomic Components

Some model specification paradigms support two types of model components, so we need to distinguish between these two types of components. The model components we have been discussing thus far are *atomic*

components. Atomic Components (AC's) are the lowest level model components of a model. Each AC in a model is an encapsulated entity that is required to have a causal behavior specification which specifies the behavior of that AC. The other type of model component (which we discuss later in this section) is a *coupled* component. A coupled component is an encapsulated model component formed by coupling together existing model components to form a "new" model component. A coupled component has a coupling specification instead of a causal behavior specification.

The causal behavior specification of an AC specifies the component's response to stimuli. The stimuli includes the arrival of messages and the passage of simulation time. The response includes the sending and receiving of messages as well as the transformation of the internal state of the component.

It is up to the modeler to determine the set of AC's that comprise a particular model. The modeler chooses an "approach" or "view" from which to model the system by analyzing the system to be modeled and the goals or purpose of the study. The set of AC's that make up any particular model are model dependent. Although there are no strict guidelines for what the AC's of a model should be, often the individual model components directly represent (model) some system component.

The modeler chooses how to model the system and thus determines the AC's of a model. The only requirement of an AC in a model is that each AC is an encapsulated entity that has a causal behavior specification and communicates with the outside world solely via message passing through a strictly defined interface consisting of a set of output ports and a set of input ports.

Since the world view we are using consists of a set of encapsulated interacting model components, it is natural to think of the model as consisting of a set of concurrently operating components which interact with each other via message passing. From this view there is a natural model parallelism where each AC in the model generally has it's own *thread of control* driving its causal behavior. (Note that thread of control as used here is *model parallelism*, a modeling construct rather than an execution construct and should not be confused with the software engineering usage which we refer to in this discussion as *computational parallelism*). If the modeller discovers a situation where there seems to be a natural model parallelism within an AC, than that component may be a candidate for splitting into two or more AC's, each with its own thread of control.

In a model specified in this manner there is a natural mapping to parallel execution using as many parallel threads of control (*computational parallelism*) as there are AC's in the model. The only limitation of parallel execution is the synchronization required for message passing between components. Synchronization is necessary to insure correct operation of the model. Note that the *model parallelism* concept of one thread of control per AC does not preclude additional *computational parallelism* within an AC during simulation execution (e.g., several calculations can be performed in parallel as part of a single control state transition).

We specify the causal behavior for an AC using a specification structure called an Atomic Component Causal Behavior Specification Structure (ACCBSS). Each AC in a model has its own causal behavior and therefore its own ACCBSS. An ACCBSS is an abstract specification concept which can be defined using any

of several concrete specification structures. Since we have not stated a particular paradigm for specification of an AC's causal behavior, any method of specification that does not violate either the encapsulation of the AC's or the message passing paradigm can be used.

An ACCBSS can use a systems theoretic specification such Zeigler's DEVS[Zei84a], a simulation or programming language specification such as Sim++[GB89] or GPSS[CGLM93], or a graphical specification such as a Petri Net. It is even possible to specify the behavior of the AC's of a model using different types of ACCBSS's for different AC's within the same model. Thus we could have a model where the behavior of some components are specified via Petri Nets, others are DEVS-Scheme atomic components, and still other components are specified using a general purpose programming language.

A difficulty that can arise with an ACCBSS is that the causal behavior specification of more complex AC's can be difficult to express in some paradigms. Some paradigms allow the behavior of an AC to be decomposed into a set of disjoint partial behaviors, each of which is specified via a Partial Behavior Specification (PBS). A paradigm which supports encapsulation of the PBS's may also facilitate the reuse of sub-AC structures (PBS's) across different AC's or even within the same AC.

If the model paradigm used for the ACCBSS supports a recursive decomposition of AC behavior then we have an even more powerful structure for modeling complex behavior. In this case we have a hierarchical behavior specification and the ACCBSS is organized as a tree structure called the ACCBSS tree. Each node of the ACCBSS tree is specified via a PBS which encloses the partial behavior for the AC specified by the subtree of the node. Paradigms which support hierarchical causal behavior specification include Hierarchical Petri Nets[Alp91], RESQ[CGLM93], and our hierarchical extension of CFG Models in Section 5.

3.2 Coupling of Components

Each AC generally has a set of output ports to which it can send messages, and a set of input ports from which it can receive messages. When using encapsulated AC's, not only are the internals of the AC's hidden from the outside view, but the outside is also hidden from the internal view of the AC's. Thus when an AC sends a message to an output port it has no knowledge of who the receivers of the message will be or what transformations, if any, will be performed on the message while it is in transit. Similarly when an AC receives a message on an input port it has no knowledge of which AC sent the message or what, if any, transformations were performed on the message while it was in transit.

The second type of specification in a complete model specification is the Component and Interconnection Specification Structure (CISS). The CISS of a model specifies the model components of a model and their *coupling* (i.e., what are the components that make up the model and how can these components interact). The interaction of components is defined by the routing and transformation of intercomponent messages. The CISS specifies the structure of the model outside of the AC's of the model.

We should be aware that in a simulation execution a message may have a wall clock "time in transit"

between the time the message is sent and the time it is received that varies based on the computer system (and the load on it) that the simulation is being executed on. However, in the conceptual model where we use "simulation" time rather than wall clock time, the time in transit for a message is zero (i.e., messages from the sending model component arrive at the receiving model component in zero simulation time). In the case where a simulation time delay for message transmission is required as part of the model, the modeler explicitly specifies the time delay as part of the model specification. We use simulation time with instantaneous transfer of messages in order to generate deterministic behavior for a model. (Even though DES models commonly use random variates, the random variates are generated using deterministic algorithms from streams of *pseudo*-random numbers. So repeating an experiment using identical streams of pseudo-random numbers will generate identical model behavior.) Determinism is a required property for the support of model verification and validation. Handling message passing so that zero simulation time passes for a message transfer is an implementation issue rather than a modeling issue. Having stated this, we will be concerned with only the model aspect of message transmission in which message transit time is zero.

There are several issues with respect to how message passing between AC's is handled in a model specification. One issue which we have already discussed is whether the model uses active receiver or passive receiver message passing. Another issue is the cardinality of port connections allowed. Does the model paradigm allow port to port connections to be *one-to-one* (i.e., each output port connected to exactly one input port and each input port is connected to exactly one output port), *one-to-many*, *many-to-one*, or *many-to-many*? A third issue is whether or not transformations on messages are allowed between the time they are sent to an output port and the time they arrive on an input port. A fourth issue is whether the modeling paradigm allows ports in a model which are not connected to other ports. Although a port with no connection may initially seem unlikely (why specify a port that is not used?), it is a possibility in such cases where a model component is designed for reuse in different models or in a different place in the same model.

One example of a coupling paradigm is Zeigler's DEVS-Scheme[Zei90], which is an object-oriented paradigm. In DEVS-Scheme each model component has a single output port to which it can send messages and a single input port from which it can receive messages. This paradigm uses passive receiver message passing, *many-to-many* port coupling, and the transformation of messages. In DEVS-Scheme when a component sends a message to an output port, the message is intercepted by an associated *coordinator* element. The coordinator looks in a connectivity table to see which ports the sending port is connected to and what transformations need to be performed on the message. After performing the appropriate transformations on the message, the coordinator forwards copies of the message (with the correct transformations) to the appropriate (connected) ports. In schemes such as this where each component has a single input port, all messages arrive at the single input port of a component and the receiving component must use a message field to differentiate types of messages.

Another example of a coupling paradigm is the interconnection graph in Cota and Sargent's CFG

Models[CS90a]. In CFG Models each model component has a set of output ports and a set of input ports. The CFG Model paradigm uses active receiver message passing, allows only *one-to-one* coupling cardinality of ports, and does not allow transformation of messages. Because CFG Models use active receiver message passing, incoming messages are queued until the receiving component chooses to receive them. Since different types of messages are generally sent over different channels, and thus arrive on different input ports, a receiving component can choose to temporarily ignore messages arriving on certain input ports when ignoring those messages does not affect the behavior of the component.

It should be noted that some issues that may seem overly restrictive are not necessarily limiting but simply require a different approach. For example, a paradigm that allows only *one-to-one* connections and no transformations on messages can model systems that naturally have *many-to-many* connections and transformations of messages by adding additional AC's between the existing AC's whose sole purpose is to merge, transform, replicate, and distribute message streams.

Thus far we have discussed how each AC in a model has an ACCBSS which specifies its causal behavior and how each model has a CISS which specifies how the AC's can interact (connections and transformations). In general, a complete model specification has a single CISS which specifies everything outside the AC's of the model and a set of ACCBSS's, one ACCBSS for each AC in the model. This specification can be represented using the structure: Model $M = \langle \text{CISS}, \{\text{ACCBSS}\} \rangle$. We note that as the number of AC's in a model increases and/or the complexity of the interactions of the AC's increases, the CISS can become very complex. Support for a hierarchical CISS is one way of addressing this complexity.

3.3 Coupled Components and Hierarchical Coupling

Some model representations allow a second type of model component, a *coupled* component, in addition to AC's. A coupled component is an encapsulated model component formed by coupling existing model components together and "drawing" an encapsulation boundary around this coupling of model components thus forming a "new" component. Drawing an encapsulation boundary around a set of components will "cut" connections between components inside the new encapsulation boundary and those components outside the encapsulation boundary. Wherever a connection crosses the coupled component encapsulation boundary we add a port to the coupled component. We add an *output* port to the coupled component when the connection is from an output port of a component that resides inside the encapsulation boundary to an input port of a component that resides outside the encapsulation boundary. Similarly, we add an *input* port to the coupled component when the connection is from an output port of a component that resides outside the encapsulation boundary to an input port of a component that resides inside the encapsulation boundary.

A coupled component has an internal view and an external view. In Figure 2 we show a coupled component **A** which contains and specifies coupling for two components **B** and **C**. Figure 2 gives us both the internal and external views of **A** and the external views of **B** and **C**. (Note that from the external view it is impossible

to determine if **B** and **C** are coupled or atomic components.) From the external view of **A** we see that **A** has two input ports (α and β) and two output ports (γ and δ). Since components are encapsulated, from the external view **A** is a black box specified by the encapsulation boundary with only the four ports visible.

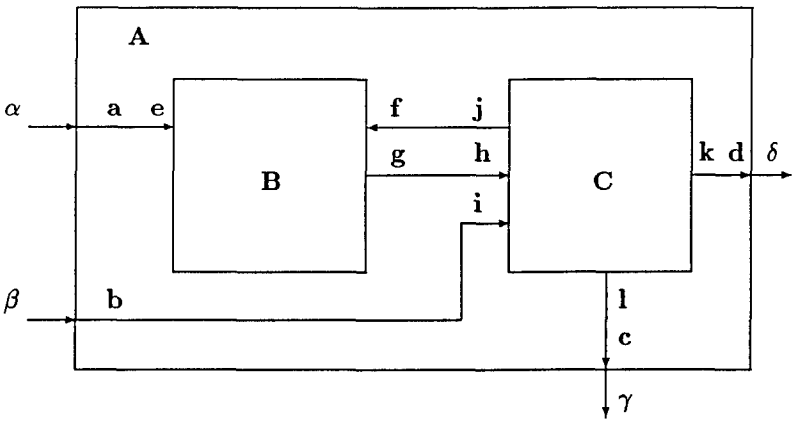


Figure 2: Coupled Component

The internal view of a model component is the view from a point inside the component but outside all enclosed subcomponents, in this case inside **A**, but outside **B** and **C**. The ports which are *visible* from this view include all ports which cross the encapsulation boundaries of components **A**, **B**, and **C**. From the internal view of **A**, we see another view of the same four ports cross **A**'s encapsulation boundary that crossed the boundary from the external view (i.e., port α from the external view and port **a** from the internal view are the same port and so on). However, from the internal view of a component we call the ports that cross the encapsulation boundary of the enclosing component *external* ports. Thus from the internal view **A** has two *external input* ports (**a** and **b**) and two *external output* ports (**c** and **d**). In addition to the four external ports already mentioned, from the internal view of component **A** there are four input ports visible: **e**, **f**, **h**, and **i** and four output ports visible: **g**, **j**, **k**, and **l**. We distinguish external ports from other ports to make clear the direction of message flow through the ports. From either the internal view or the external view of a model component we have a set of ports that are visible (not hidden by an encapsulation boundary). We can determine the direction of message flow through any visible port by a simple rule. From either view (internal or external) of a component, observing all ports visible from that view, messages flow toward the observer from output ports and external input ports and away from the observer to input ports and external output ports. (Note that external ports exist only from the internal view of a component.)

Coupling between ports of components contained within a coupled component is referred to as *internal coupling*. Coupling between components within a coupled component and components outside of (external ports of) a coupled component is referred to as *external coupling*. We note that it is also possible to have direct coupling between an external input port and an external output port of a coupled component which means that messages are simply routed through the component with no effect on the internals of the coupled

component.

Since a coupled component is an encapsulated entity with a set of input ports and a set of output ports, without looking inside a model component it is impossible to distinguish a coupled component from an AC. Thus coupled components can be used in model construction in the same manner as AC's. This ability to couple existing model components together to form new model components is known as *closure under coupling*[Zei84b]. This closure under coupling property, along with the encapsulation of model components, is sufficient to allow hierarchical model specification[Zei90]. This is one of the two types of hierarchical model specification. Zeigler's DEVS-Scheme[Zei90] is a model representation whose CISS supports hierarchical component specification in this manner.

In a CISS which supports hierarchical model specification, the CISS can be viewed as a rooted tree (called a CISS tree) where the internal nodes of the tree represent coupled components and the leaf nodes of the tree represent AC's. A modeler specifies each AC via an ACCBSS, but there must also be a method for specifying the coupling definition for each coupled component. The specification of the internals of each coupled component is accomplished via a Coupled Component Specification (CCS). Thus each internal node in the CISS tree is a coupled component (defined via a CCS) which specifies the components contained within that component (child nodes in the CISS tree) and the coupling for those components. Although we can view the CISS as a rooted tree in which the CISS tree shows hierarchical relationships between components, the CISS tree does not show the port-to-port coupling information. The coupling information is contained in the individual CCS's. Thus a CISS is completely specified via a set of CCS's, one CCS for each internal node of the CISS tree. Note that a non-hierarchical CISS is a simply a special case of a hierarchical CISS in which the CISS tree has a depth of one. In this case there is a single coupled component (specified via a CCS) that specifies the coupling for all the other components (AC's) of the model.

We have stated that AC's have a causal behavior specification (and thus a behavior). Coupled components have a coupling specification instead of a causal behavior specification. Thus in Figure 1 coupled components are part of the coupling specification (above the "table"). Although coupled components do not have a causal behavior specification, if we observe the external view of a coupled component during simulation of the model we can observe behavior in the form of the sending of messages to output ports in response to the receiving of messages on input ports and the passage of simulation time. Thus, from the external view, in simulation execution, as well as in model construction, it is impossible to distinguish a coupled component from an AC.

However, since a coupled component represents an internal node of the CISS tree and leaf nodes of the CISS tree represent AC's, if we apply a recursive decomposition to any coupled component in the model we eventually arrive at a point where all remaining components are AC's. After this decomposition we see that the number of AC's (recursively) contained within any coupled component is equal to the number of leaf nodes of the CISS subtree where the original coupled component is the root node of the subtree. Thus a coupled component has a behavior which has as many threads of control (model parallelism) as the number of enclosed AC's. The (observed) behavior of a coupled component is a function of the coupling and behaviors

of the subcomponents that make up the coupled component. Applied in a recursive manner, the behavior of a coupled component is a function of the coupling and behaviors of the components of the subtree of the CISS tree of which that coupled component is the root node. Thus the behavior of the coupled component at the root of the CISS tree is the behavior of the entire model.

We have already selected a sufficiently narrowed paradigm for the port coupling (encapsulated AC's, message passing, and hierarchical coupling) that it is possible to give a formal specification for a CISS for this paradigm. In the following subsection we specify a formal CISS which supports this coupling paradigm. This CISS is sufficiently general that the CISS used by Zeigler's DEVS-Scheme[Zei90], the CISS used by Cota and Sargent's CFG Models[CS90a], and the CISS we use for our hierarchical extension to CFG Models, HCFG Models, are simply special cases (restrictions) of the formal CISS.

3.4 A Formal CISS

Although we can view the CISS as a rooted tree, the CISS is completely specified via a set of CCS's, one CCS for each internal node (leaf nodes are AC's) of the CISS tree. Thus to formally specify a CISS we need only a formal structure for a CCS, and a set of CCS's specified via this formal structure.

Components (coupled and atomic) have both an external view and an internal view. From the external view, a component is a "black box" with a set of input ports to which messages can be sent to the component, and a set of output ports from which from which messages can be received from the component. Both sets of ports are ordered sets which allows us to distinguish individual ports. The formal structure for the external view of a component M_j is

$$M_j = \langle I_j, O_j \rangle \qquad \text{where}$$

- I_j is the ordered set of all input ports of model component M_j.
- O_j is the ordered set of all output ports of model component M_j.

Although we have given a specification for the external view of a component, the external view of any model component can be derived from the component's internal view.

A coupled component M_j^C is a model component that is composed of a coupling of model subcomponents. These subcomponents are either AC's and/or other coupled components. Each coupled component is represented by a CCS which specifies the internal view of that coupled component and each AC is represented by an ACCBSS which specifies its causal behavior. The internal view of a coupled component M_j^C specifies a set of subcomponents $\mathcal{M}_j = \{M_i\}$ of M_j^C and a port-to-port coupling Γ_j of ports. These ports include the external ports of M_j^C and the ports of the subcomponents $\{M_i\}$.

Formally, for a coupled model component M_j^C, the CCS of M_j^C is a structure

$$M_j^C = \langle \mathcal{M}_j, \Gamma_j \rangle \qquad \text{where}$$

- $\mathcal{M}_j = \{M_i\}$ is the set of subcomponents, M_i, of M_j^C.
- Γ_j is a port to port coupling from Ω_j to Υ_j where each ordered pair $(\alpha, \beta) \in \Gamma_j$ has an associated message transformation function (which may be a *null* (do nothing/do not modify) transformation) that is applied to every message received from port α before it is delivered to port β where
  - $\Omega_j = I_j^E \bigcup (\bigcup_i O_i)$ is a set of ports where
    - I_j^E is the set of external input ports of M_j^C.
    - O_i is the set of output ports of subcomponent M_i of M_j^C so $\bigcup_i O_i$ is the union of the sets of output ports of all the subcomponents of M_j^C.
  - $\Upsilon_j = O_j^E \bigcup (\bigcup_i I_i)$ is a set of ports where
    - O_j^E is the set of external output ports of M_j^C.
    - I_i is the set of input ports of subcomponent M_i of M_j^C.

If we have a CCS where $I_j^E = \emptyset$ (the empty set) and $O_j^E = \emptyset$, then we have a self contained coupled component that does not interact with the outside world. In a CISS, there is exactly one component of this type, the coupled component at the root of the CISS tree. (If there is more than one CCS in the CISS tree with this property, then we have multiple independent models which have no interaction whatsoever. If we have no CCS with this property, then our CISS is not completely specified because the coupled component at the root of the CISS tree is trying to communicate with a non-existent entity outside the CISS tree.)

Now that we have a formal specification for a CISS, we can see that Ziegler's DEVS-Scheme[Zei90] CISS is a special case where each component (except for the root node of the CISS tree) has a single input port and a single output port and coupling that allows non*null* message transformation functions. Cota and Sargent's CFG Model CISS, i.e., their interconnection graph, is a special case of this CISS which is discussed in section 4, and our HCFG Model extension of CFG Models is another special case of this CISS which is discussed in section 5.

4 Control Flow Graph Models

Control Flow Graph (CFG) Models are a DES model representation developed by Cota and Sargent[CS90a]. The CFG Model representation is based on the modified version of the process interaction world view and active receiver message passing as described in section 3.

The CFG Models were designed not as a modeling language, but as a model representation which makes information useful for parallel simulation explicit. Algorithms have been developed[CS90c] for CFG Models which allow models using the CFG Model representation to be executed on either sequential or parallel computer architectures.

A CFG Model consists of a set of AC's (Atomic Components), (also known as Logical Processes or LP's) which interact with each other via active receiver message passing. All model components in a CFG Model are atomic. Each AC in a CFG Model is a modular entity which represents a system component in the modeled system.

In a CFG Model the ACCBSS is defined via a structure called a CFG and the CISS is defined via a structure called an Interconnection Graph.

4.1 ACCBSS

A CFG (which specifies the causal behavior of an AC) consists of a set of input ports from which messages can be received, a set of output ports to which messages can be sent, a set of variables for maintaining local state (including a local simulation clock), and an augmented directed graph. In the augmented directed graph, the nodes are control states and the edges represent the potential control state transitions of the AC. Each edge in a CFG has associated with it a condition, a priority, and an event.

The condition on an edge specifies under what conditions that edge can be considered as a potential next edge for traversal. Conditions are one of three types. A *non-empty* input port condition specifies an input port of the atomic component, and is true if there is an unreceived message available on that input port. A *boolean* condition specifies a boolean expression which may include tests of the local variables of the atomic component. A *time-delay* condition specifies a time delay, and the condition becomes true after the passage of the specified local simulation time. An edge traversal time (simulation time at which the edge can be traversed) can be computed from an edge's condition and the current AC state. This traversal time may require recomputation in some cases when new messages arrive on an input port. When selecting an edge for traversal, we choose an edge leaving the control state that has the smallest traversal time.

Sometimes more than one edge leaving a control state has the same minimum traversal time and we then have a set of edges which are candidates for traversal. In this case the priority's of the edges are used to select the edge for traversal from this set of candidates. The priorities on the set of edges leaving each control state are required to be unique so that only one edge is selected.

The event associated with each edge specifies a state transformation for the component. The actions performed as part of a state transformation can include receiving a message from an input port, sending messages to output ports, and modifying the values of local variables.

A CFG always has exactly one "current" control state (or point of control). The basic operational semantics of a CFG is to repeatedly select one of the edges leaving the current control state for traversal. At times an AC may have to delay the selection of this edge in order to allow for the arrival of new messages on the component's input ports which could influence the selection of this next edge. Once the next edge to be traversed is selected, the local simulation clock is advanced to the traversal time of the selected edge. We then traverse the edge, executing the state transformation associated with it, and setting the control state

that the selected edge is incident on as the "new" control state (or point of control). This basic operation is repeated until a simulation termination condition is met.

Synchronization between AC's is accomplished via time-stamped messages. Each message that a component sends to an output port has a timestamp which is equal to the local simulation clock of the sending component at the time the message is sent. Note that setting the timestamp of a message to the time sent differs from the classical parallel DES paradigm in which the message timestamp is set to the time that the message is to be received. Since the local simulation clock is non-decreasing, all messages sent by a component have non-decreasing timestamps. All messages sent over a specific channel are received from that channel in the order in which they were sent; thus messages received from each channel also have non-decreasing timestamps.

4.2 CISS

The CISS of a CFG Model is called an Interconnection Graph(IG). The IG specifies the coupling of component ports. Since there are no coupled components in a CFG Model, the CISS can be specified with a single CCS (Coupled Component Specification). In a CFG Model the coupling of ports is a special case of the CISS described in section 3. An IG has a *one-to-one* port mapping and no message transformations. Thus the coupling in an IG can be specified as an invertible function from the union of the sets of output ports of all the AC's of the model to the union of the sets of input ports of all the AC's of the model. Recall that in CFG Models the connections between ports are referred to as channels and we say that messages flow over channels. Different types of messages are generally sent over different channels; thus, there may be multiple channels connecting output ports of one component to input ports of another component with each channel carrying a different type of message. Since CFG Models use active receiver message passing, messages queue until received. Operationally it makes no difference if messages queue on channels or on ports. Cota and Sargent specify that messages queue on channels while in our hierarchical extension of CFG Models in section [ref] we have messages queue on input ports. This difference in the location of message queueing does not affect model operation but it does affect how the total model state is specified.

4.3 Algorithms

Cota and Sargent[CS90c] have developed a set of simulation algorithms for the execution of CFG Models. These algorithms allow CFG Models to be executed on different types of computer architectures. The primary advantage of this is the capability to execute a model on either a sequential processor system or a parallel/distributed multi-processor system without the modeler adding addition information to the model [CS89], as is generally required in most existing parallel DES paradigms.

A CFG Model can be simulated on a sequential architecture computer using a synchronous algorithm or an asynchronous algorithm. The synchronous algorithm executes all events (state transitions) in all atomic

components in strict time order. The asynchronous algorithm may execute some events out of time order in cases where the execution order does not affect the simulation result. This "out of order" execution can in some cases provide an execution speedup by eliminating some unnecessary event list operations.

A CFG Model can be executed on a parallel/distributed architecture system using one of their conservative algorithms, an optimistic algorithm, or an Optimistic only When Necessary (OWN) algorithm.

In a conservative parallel DES, events are executed for an AC only when it can be guaranteed that the execution of that event will result in correct behavior of the AC. The speedup achieved with conservative parallel DES is highly dependent on the amount of lookahead that can be achieved[Fuj90]. For most model representations, this requires that the modeler explicitly find and build knowledge useful for lookahead computation into the model. Explicitly building this lookahead information into the model generally requires an understanding of the algorithm used for parallel simulation [Nic88, Nic90, LL89, Wag91]. In contrast, one of the fundamentals of CFG Models is that information needed for lookahead computation is made explicit. Thus the parallel DES algorithms for CFG Models can do "automatic lookahead" [CS90b] computation, which allows the modeler to achieve execution speedup on parallel/distributed architecture machines without being concerned with any aspects of parallel DES.

Optimistic algorithms allow AC's to save state and continue to execute events, rolling back to a previously saved state if it is discovered that a previously executed event was incorrect (i.e., based on an assumption which later proved false). The optimistic algorithm used for CFG Models improves the efficiency of the Time Warp algorithm [CS92]] by reducing the frequency and distance of rollbacks.

The OWN algorithm allows an AC to execute in the conservative mode, removing the need to save state, until such time as the AC reaches a point where it would block execution waiting on information from another AC. At the point where an AC would block and otherwise sit idle, the AC switches into optimistic mode and continues operation in optimistic mode. An AC using the OWN algorithm that has switched into optimistic mode attempts to switch back into conservative mode at a later time.

The set of algorithms available for CFG Models makes the CFG Model representation highly desirable from a simulation execution viewpoint (particularly with respect to parallel/distributed execution). Although it is possible to develop models using CFG Models, CFG Models were not designed for that purpose. It is desirable to have a modeling language or model representation oriented toward model development which could take advantage of the existing algorithms for CFG Model execution. We present such a model representation in the form of Hierarchical CFG Models in the next section.

5 Hierarchical Control Flow Graph Models

Models specified using the CFG Model representation can be executed on either sequential architectures or parallel/distributed architectures using existing algorithms. However, it can be difficult to develop a CFG model when there are a large number of AC's and/or the AC's are large or complex. To address this, we

have developed a hierarchical model representation based on CFG Models which can be algorithmically mapped into equivalent CFG Models, thus allowing the use of the existing execution algorithms. As noted in section 3, models based on encapsulated AC's and message passing require two types of model specifications to completely specify a model: an ACCBSS to specify the causal behavior of each AC and a CISS to specify coupling of model components. Our hierarchical extension of CFG Models, HCFG (Hierarchical Control Flow Graph) Models, consists of two types of extensions that can be used independently of each other.

One extension, the Hierarchical Interconnection Graph (HIG), provides support for a hierarchical CISS by supporting encapsulated coupled components as discussed in section 3. The HIG tree is a special case of the CISS tree of section 3. A HIG can be algorithmically transformed into an equivalent (CFG Model) IG. The other extension, a Hierarchical Control Flow Graph (HCFG), provides support for a hierarchical ACCBSS through the use of macro control states. An HCFG can be algorithmically transformed into an equivalent CFG. Thus an HCFG Model can be algorithmically transformed into an equivalent CFG Model. We show that a CFG Model is a special case of an HCFG Model and thus a CFG Model can be specified using the HCFG Model specification.

We have two primary objectives for the HCFG Model representation. One objective is to facilitate model development to make it easier to develop, maintain, and reuse model elements (components in a CISS and macro control states in an ACCBSS). This objective is addressed by supporting encapsulation of model elements to enable reuse of the model elements and hierarchical structures for model specification as an aid in managing model complexity.

The second primary objective for HCFG Models is to support flexible and efficient execution of models. This objective is addressed by providing an algorithmic mapping from an HCFG Model into an equivalent CFG Model in order to execute the model using the algorithms developed for CFG Models. Algorithms for transforming an HCFG Model into an equivalent CFG Model are given in section 6.

The HCFG Model extension to CFG Models also satisfies the requirements for a new modeling paradigm as specified by Sargent [Sar92] which include a theoretical foundation, hierarchical modeling capability, support for model component reuse, ease of modeling, and support for simulation execution on different computer architectures.

In the remainder of this section we first specify the HCFG Model CISS and show a simple example using the CISS. Then we describe the HCFG Model ACCBSS, providing formal specification and operational semantics. Next we present a brief example which illustrates the modeler's view of an HCFG.

5.1 Hierarchical Interconnection Graph

In an HCFG Model the HIG is a hierarchical extension of a CFG Model IG which allows coupled components and thus has a hierarchical coupling specification. The CISS tree in Section 3 becomes the "HIG tree" where the internal nodes of the tree represent coupled components and the leaf nodes represent AC's. The coupled

components are specified via a CCS (Coupled Component Specification) which specifies the internals of a coupled component. The HIG is a special case of the hierarchical CISS specified in section 3 in that a CCS specifies a *one-to-one* mapping for port coupling and has no message transformations. If a HIG tree has a depth of one (i.e, a root node and the rest of the nodes are leaves), then the CCS for the root node of the HIG is equivalent to a CFG Model IG. Thus an IG is a special case of a HIG which can be specified via a single CCS.

Thus the formal structure for the HIG is the same as the formal structure for a CISS given in section [ref], but with restrictions that only *one-to-one* port coupling is allowed and no message transformations are allowed.

5.2 HIG Example

We use a simple production system that can be modeled as an open queueing network to illustrate the HIG of an HCFG Model using CCS structures.

In this model we have two service centers, each with a finite input queue, and a single inspection station with an infinite queue. As jobs arrive, they are routed to one of the two service centers, or turned away if the system is full, i.e., the input queues of both service centers are full. Jobs leaving a service center are sent to the inspection station. Following inspection, jobs leave the system. Additional model details are as follows:

- Jobs, which arrive randomly, belong to one of two classes, high priority or normal. Upon arrival, jobs are routed to the service center with the shortest input queue. If both queues are the same length, the first service center is selected. If both input queues are full, the job is turned away (discarded).
- Two identical service centers, each consisting of two tandem servers, service jobs. The first server follows a priority preempt queue discipline when selecting jobs from its input queue. When a job finishes service at the first server it is sent to the second server (which has no queue). The first server blocks if the second server is busy when a job finishes service at the first server.
- Jobs leaving a service center are sent to the inspection station. After inspection jobs leave the system.

Starting with a top down development of the model, we can break the model into three components: a **Source** which simulates the arrival of new jobs, the **System** which processes the jobs, and a **Sink** which removes jobs from the model as they finish processing. The CCS for the root node of the CISS tree, labeled **TOP**, is shown in Figure 3. The **Source** component uses two types of messages to distinguish the two classes of jobs. Messages representing high priority jobs are sent over the channel labeled **hi** and normal jobs are sent over the channel labeled **lo**.

The **source** which generates job arrivals and the **sink** which removes completed jobs from the model are simple and thus are modeled as AC's. The **system** is sufficiently complex (and contains multiple threads of

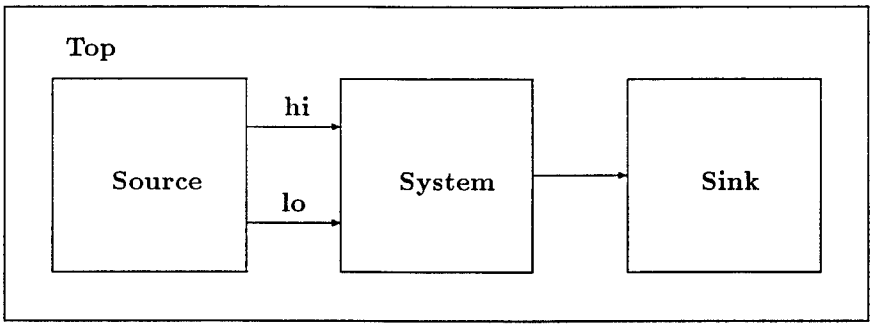


Figure 3: Coupled Component Specification for **TOP**

control) that it is modeled as a coupled component which is then further decomposed.

We decompose the **system** component into four subcomponents, a **router** which routes jobs to a service center, two identical service centers (**svc_cntr**[i], $i \in \{1,2\}$), and an **inspector** station as shown in the **system** CCS in Figure 4. The **router** receives information on the queue lengths at the service centers via messages over the **avail_**i channels which it uses to determine which service center to send a job to. We model the **router** and the **inspector** components as AC's. We model the service centers as coupled components, thus having further decomposition. Since the service centers are identical, we model one service center and then reuse that specification for the other service center.

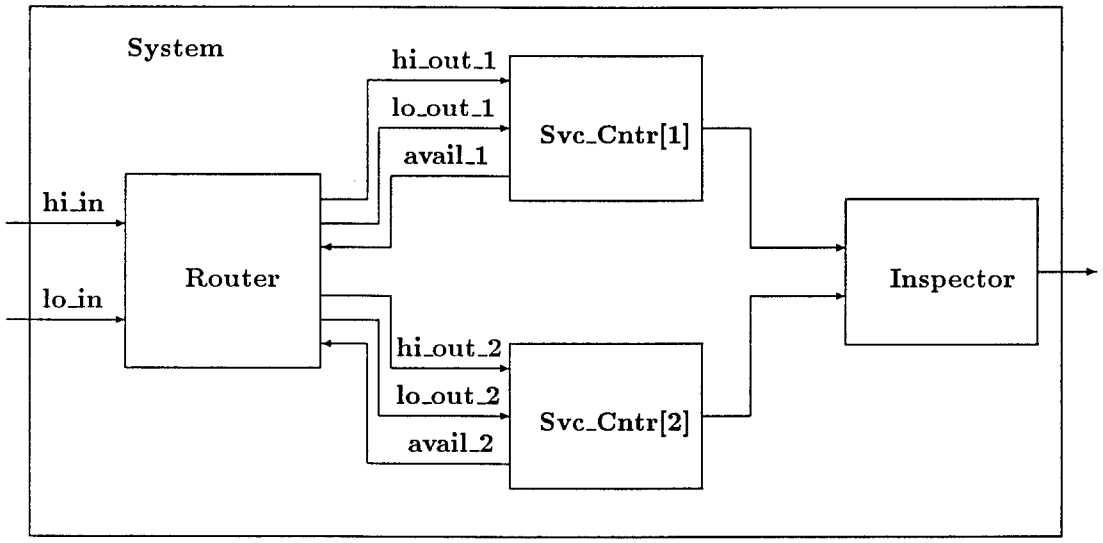


Figure 4: Coupled Component Specification for **System**

A **svc_cntr** consists of two servers **s1** and **s2** as shown in its CCS in Figure 5. The **s2_avail** channel is used to carry messages indicating the status of the **s2** server to the **s1** server. We model the servers **s1** and **s2** as AC's so there is no further decomposition in the HIG for this example.

The HIG tree for this example is shown in Figure 6. The triple vertical bar above **svc_cntr** indicates replication of the **svc_cntr** component. The (2) next to the triple vertical bar indicates the number of

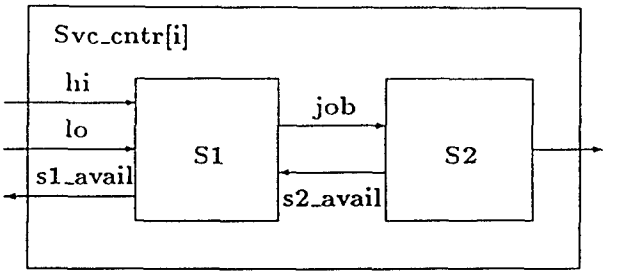


Figure 5: Coupled Component Specification for Svc_Cntr[i]

replications. The tree structure gives an overview of the components of the model. The internal nodes of the tree represent coupled components and the leaf nodes represent AC's. The tree structure does not show coupling information. For coupling information we must consult the individual CCS's.

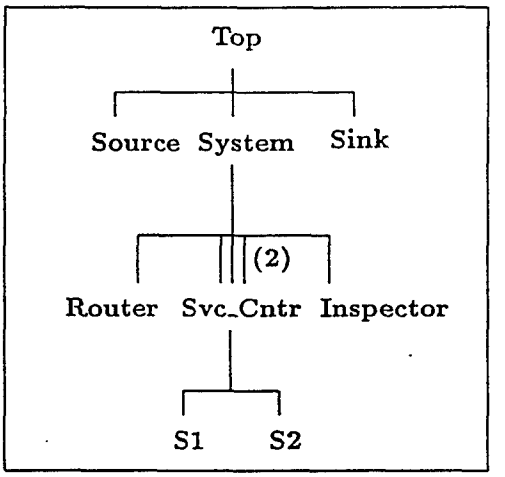


Figure 6: Rooted Tree Representation

5.3 HCFG

The AC's, the lowest level components of an HCFG Model, each have a causal behavior specification (AC-CBSS). In a CFG Model, the ACCBSS is a CFG. For AC's with relatively simple behaviors, CFG's for those AC's can be developed easily. However, it can be difficult to develop a CFG when AC's are large and complex. We note three specific areas of concern in which development of a CFG may become troublesome.

One concern with using CFG's as an ACCBSS is that the causal behavior of an entire AC is specified via a single CFG (which has a single augmented directed graph). As the behavior of an AC becomes more complex, the complexity of the CFG required to model that behavior can be difficult to generate. For example, a large number of control states and a large number of edges may be needed to model a complex behavior.

A second concern with CFG's is that there is a single name scope for all the variables of the AC. This

means that the modeler must give unique names to unrelated variables which are used in different parts (e.g., the *condition* and *event* specifications of edges) of the CFG . This name scope concern of CFG's is analogous to a programming language in which all variables are global variables. Having a single crowded name scope can be an inconvenience to the modeler and can lead to difficult to detect errors in the CFG specification.

A third concern with CFG's is that although the CFG Model representation supports the reuse of coupled and AC's, there is no support for the reuse of model elements smaller than (within) an AC.

We address these concerns in HCFG Models with a hierarchical ACCBSS called an HCFG (Hierarchical Control Flow Graph). We use a method of "State Abstraction" similar to that in [dLF93] in which we specify AC state and behavior using a hierarchical structure. We can view an HCFG as a tree structure called the "HCFG tree" in which the nodes of the HCFG tree correspond to *abstract* or *macro* control states of the AC. (The HCFG tree is the ACCBSS tree mentioned in section 3.) Macro Control States (MCS's — pronounced "em-see-ess") encapsulate a partial behavior of a model component. Specifically, an MCS encapsulates one or more control states and/or MCS's and their associated edges (note the recursive definition). Edges originating at encapsulated control states have the set of attributes (condition, priority, and event). A Macro Control State Specification (MaCSS — pronounced "Max") is a formal specification structure that specifies the internals of each MCS. Where an HCFG specifies the causal behavior for an entire AC, each node of an HCFG tree (except for the root node) generally specifies only a partial behavior of an AC (the behavior expressed within a single MCS). Note that the MCS at the root of the HCFG tree encloses the entire (rather than a partial) behavior of the AC. For each node, say G_j, in the HCFG tree, the number of child nodes of G_j in the HCFG tree and the number of MCS's in the MaCSS for G_j are the same (an MCS in a MaCSS indicates a child node in the HCFG tree). A MaCSS is the specification for an MCS, but where the meaning is clear we use MaCSS and MCS interchangeably.

According to Zeigler[Zei90], a hierarchical model can be executed either using a hierarchical simulator, or a nonhierarchical simulator by first flattening the hierarchical model to form an equivalent nonhierarchical model. In this section we discuss the hierarchical operation of an HCFG. (Cota and Sargent's description of the operation of a CFG in [CS90a] is used if the HCFG is first flattened into a CFG. Flattening algorithms are given in section 6.)

Recall that each AC has a single thread (or point) of control. Thus when one views the HCFG tree of an AC, the current control node (point of control) always resides in exactly one node of the HCFG tree (and thus exactly one MCS) at any point in time. Inside an MCS we find a structure similar to a CFG but with some important extensions.

Recall that in a CFG the single point of control moves from control state to control state and always resides in one of the control states. In a MaCSS the point of control can move not only to a control state, but it can also enter an MCS, or it can leave the current MCS entirely. The control states in an MCS are identical to those in a CFG. When the point of control in an MCS enters an enclosed MCS the point of

control is actually moving from the current node in the HCFG tree to the child node in the HCFG tree represented by the MCS. In this case control is passed to the MaCSS that represents that child node in the HCFG tree. When the point of control leaves an MCS, it moves up one level in the HCFG tree to the parent node of the current node in the HCFG tree and the point of control passes to the MCS that represents that parent node. Thus as the point of control moves from control state to control state (as discussed in section 4), it may move up and/or down the HCFG tree as it goes through one or more MCS's.

Looking at the HCFG tree we see two special cases of MCS's. We see that MCS's representing leaf nodes of the HCFG tree contain no other MCS's because the presence of a MCS inside another MCS indicates a child node in the HCFG tree. We also see that the point of control can never move outside the root node of the HCFG tree because the root node (which has no parent) encloses the total behavior of the AC. We note that a modeler can choose to model in such a manner that all control states are contained within the leaf nodes of HCFG tree. In this case the internal nodes of the HCFG tree contain only MCS's. Also note that if the HCFG tree consists of only a single node then the HCFG is equivalent to a CFG. Thus we can use a MaCSS structure to represent a CFG since a MaCSS structure contains all the information required to specify a CFG.

Another important aspect of MaCSS's is the encapsulation and the use of "handles" to access information. The encapsulation enforces a strict interface between levels of the HCFG tree which provides each node in the HCFG tree with it's own name space for variables and ports and helps prevent an MCS from accidentally accessing information that it should not access (information hiding)[Str91]. Handles are a delayed binding of information in a manner similar to the concept of formal versus actual parameters in a subroutine call of a programming language which uses "call by reference". Thus instead of accessing a particular variable of type $\mathcal{T}$, we access a variable of type $\mathcal{T}$ via a handle $\mathcal{H}$. The only difference is that we no longer know where the actual variable is defined. This delayed binding is a static binding (does not change during model execution) for a model which is specified during model construction. Encapsulation of MaCSS's and the use of handles to access information greatly aids in the reuse of MaCSS's.

Each MaCSS is a modular (has the locality and encapsulation properties) structure which represents a partial behavior template or MCS of an AC. We say a MaCSS is only a template because in the general (i.e., non-trivial) case a MaCSS contains a list of formal parameters (handles, initialization constants, child MCS's, etc.) which need to be replaced with actual parameters. The MaCSS alone does not contain enough information to specify the behavior of a specific MCS until it is used in the context of a specific HCFG where the actual parameters are supplied. Thus a MaCSS is an inherently reusable model element which must be supplied with additional context (model) specific information prior to use.

In the following discussion we use the term *handle* to "something" to indicate the "delayed binding" of information (and a level of information indirection). If a MaCSS, say G_j, is given a handle to an output port, G_j can send messages to the output port via the port handle. However, since the actual port that the port handle refers to is determined outside of G_j, it is not possible (nor desirable) for G_j to determine the

actual port that a port handle refers to.

Some additional notes regarding handles are as follows:

- There is a distinction between
  - The name of a variable (A name has local name scope only. A variable can only be referred to directly by name in the local scope)
  - A handle to a variable (A handle to a variable can be passed down the hierarchy (HCFG tree) and the current value of that variable can be accessed by lower level MaCSS's via the handle. This is similar to the "call by reference" method of parameter passing in a programming language subroutine call.)
- There may be variables which are defined (have their local name scope) outside of the root node of the HCFG tree. Handles to these variables are passed to the MaCSS at the root node of the HCFG tree during model construction. Although these variables are outside the HCFG tree, they are still part of the HCFG.
- When we pass handles to variables down the hierarchy, there is always an *implied* "first" handle, which is a handle to *clock*, the local simulation clock. This local simulation clock is a required variable for every HCFG. A reference to the variable *clock* in any scope within an AC always refers to this same variable, which effectively becomes a global variable ***within*** the enclosing AC. Recall that each AC has its own local simulation clock.
- All handles have an initial value which we call $NULL$ which represents an ***uninitialized*** state. (There is no model requirement that all handles refer to an object (be non-$NULL$) prior to model execution. What action to take if an attempt is made to access information via a $NULL$ handle during model execution is an implementation issue).

Control flow passes from MaCSS to MaCSS through the encapsulated boundaries of the MCS's. Because of the encapsulation of MCS's we need a mechanism to represent this "hand-off" of control flow as it crosses the boundary. The mechanism we use is the concept of a ***pin***. Control flow enters or leaves an MCS through a pin, thus pins become the conduits through which control flow can enter or leave an MCS. An MCS generally has a set of pins and there is a significance as to which pin the control flow enters or leaves through because control flow paths through different pins generally terminate at different control states.

5.4 Formal Specification of a MaCSS

An HCFG Model MaCSS is a modular structure which has an external view and an internal view. From the external view a MaCSS G_j is essentially a black box with an ordered set of input pins, an ordered set of

output pins, and lists of formal parameters. Formally this is represented via a structure

⟨Pins, Initialization-Constants, Handles-to-External-Info⟩ where

- Pins are the conduits through which the point of control (control flow) can enter or leave a MaCSS. Pins consists of
  - an ordered set of input pins of G_j through which control can enter G_j
  - an ordered set of output pins of G_j through which control can leave G_j
- Initialization-Constants is an ordered list of initialization constants to be used upon instantiation of G_j
- Handles-to-External-Info consists of
  - an ordered list of handles to external variables
  - an ordered list of handles to input ports of the AC from which messages can be received
  - an ordered list of handles to output ports of the AC to which messages can be sent
  - a handle to the parent MaCSS of G_j (or $NULL$ (none) if G_j represents the root node of the HCFG tree)

The internal view of a MaCSS structure specifies the behavior of an MCS of an AC. A MaCSS structure contains sufficient information to specify a CFG so we can use the structure developed for MaCSS's to also specify CFG's.

Information from the external view of a MaCSS is accessible from the internal view by prefixing the name used in the external view with the keyword *external* (e.g., the ordered set of input pins from the external view of a MaCSS becomes the ordered set of *external* input pins from the internal view of the MaCSS).

The internal view of a MaCSS G_j is given by the structure

⟨ Local-Node-List, Local-Handles-to-Info, External-to-Self-Mappings,
Local-Mapping, Self-to-Child-Mappings, ⟩ where

- Local-Node-List consists of
  - a set of enclosed MCS's $\mathcal{G}_j = \{G_i\}$ of G_j
  - a set of local control states $\mathcal{N}_j = \{N_i\}$ of G_j
- Local-Handles-to-Info consists of
  - an ordered set of handles to local variables (the set of variables defined in G_j) (A handle to a local variable implies the definition of the local variable.)

- an ordered set of handles to external variables (variables defined outside G_j)
- an ordered set of handles to input ports of the AC
- an ordered set of handles to output ports of the AC
- a handle to the parent MaCSS of G_j (may be $NULL$ if this MaCSS is at the root of the HCFG tree)

- External-to-Self-Mappings consists of
  - a function from the state set of the external initialization constants of G_j to the state set of local variables of G_j which specifies the initial values for the local variables
  - a one-to-one mapping (binary relation) from the external ordered set of handles to external variables to the ordered set of handles to external variables
  - a one-to-one mapping from the external ordered set of handles to input ports to the ordered set of handles to input ports
  - a one-to-one mapping from the external ordered set of handles to output ports to the ordered set of handles to output ports
  - a direct mapping from the external handle to the parent MaCSS to the handle to the parent MaCSS

- The Local-Mapping consists of a control flow mapping (augmented directed graph). The control flow mapping is a mapping from the set $\mathcal{A}$ to the set $\mathcal{B}$ where
  - set $\mathcal{A}$ is the union of the three sets
    - the set of local control states $\mathcal{N}_j = \{N_i\}$ of G_j
    - the set of external input pins of G_j
    - the union of the sets of output pins of the enclosed MCS's $\{G_i\}$ of G_j
  - set $\mathcal{B}$ is the union of the three sets
    - the set of local control states $\mathcal{N}_j = \{N_i\}$ of G_j
    - the set of external output pins of G_j
    - the union of the sets of input pins of the child MaCSS's $\{G_i\}$ of G_j

  This control flow mapping has a restriction which must be satisfied. For all $\alpha \in \mathcal{A}$ and for all $\beta \in \mathcal{B}$, (α, β) is a valid element of the control flow mapping only if it satisfies the restriction that if α is a pin (as opposed to a local control state), then α can map to at most one $\beta \in \mathcal{B}$ (i.e, there can be at most one edge originating at each pin).

  There are additional attributes associated with each edge that originates at a control state. This additional information is the same information required on an edge in a CFG and consists of

- a condition – under what conditions can this edge become a candidate for traversal
- a priority – if multiple edges are simultaneously candidates for traversal, then the priorities of the edges are used to select one of the candidate edges
- an event – the transformation function associated with the traversal of an edge (may include receiving and sending of messages via the AC's ports in addition to the modification of the variable state space of the component).

Edges which originate on pins (rather than on local control states), do not have these edge attributes. We can remove all pins from any control flow path with no effect on a model's behavior. One of the effects of flattening the HCFG into a CFG is that we end up with a single MCS which contains all of the control states of the AC. Since all control states are now within the same MCS, control flow never crosses an encapsulation boundary and thus there is no longer a need for pins.

- the Self-to-Child-Mappings consist of
  - a one-to-one mapping from the union of the set of handles to external variables of G_j and the set of handles to local variables of G_j to the union of the sets of handles to external variables of the child MaCSS's $\{G_i\}$ of G_j.
  - a one-to-one mapping from the the set of handles to input ports in G_j to the union of the sets of handles to input ports of the child MaCSS's $\{G_i\}$ of G_j.
  - a one-to-one mapping from the the set of handles to output ports in G_j to the union of the sets of handles to output ports of the child MaCSS's $\{G_i\}$ of G_j.
  - a direct mapping from a handle to Self (G_j) to the handle to parent MaCSS input of each child MaCSS $\{G_i\}$ of G_j.
  - a mapping to the state sets of the ordered lists of initialization constants of the child MaCSS's $\{G_i\}$ of G_j.

5.5 Operational Semantics

The conceptual operation (abstract simulation [Zei84a]) of an HCFG is similar to that for a CFG. In this section we describe the operational semantics required for a hierarchical abstract simulation of an HCFG. Alternatively an HCFG can be first flattened into an equivalent CFG and then one can use Cota and Sargent's algorithms for CFG Models[CS90c].

In an HCFG we have a set of MaCSS's arranged in a rooted tree structure (the HCFG tree). Since an AC has a single thread of control, at any point in simulation time, exactly one of the MaCSS's (the MaCSS in which the point of control resides) is the current MaCSS. Within the current MaCSS, the point of control resides at a control state. The basic operational semantics of the current MaCSS is to repeatedly

select one of the edges leaving the current control state for traversal. The MaCSS may in some cases have to delay the selection of an outbound edge due to insufficient information (messages arrivals may influence the selection of this next edge). Once the next edge to be traversed is selected, the local simulation clock is advanced to the traversal time for the selected edge. Then the selected edge is traversed, executing the state transformation associated with it. If the selected edge terminates on a control state then that control state becomes the "new" current control state. Otherwise the selected edge is incident on a pin. In this case the point of control flows across a series of edges (and through a series of pins) until it arrives on a control state, which becomes the "new" control state. When the point of control flows through a pin into another MCS, the next control state generally resides in a different MCS than the current control state. This basic operation (of selecting an edge, executing the event associated with that edge, and traversing the edge(s) to get to the next control state) is repeated for an AC until a simulation termination condition is met.

A simple example is presented in the next subsection as an aid in understanding HCFG's and MaCSS's

5.6 HCFG Example

We use a modification of the flexible manufacturing system example presented in [Sar88] to illustrate the modeler's view of an HCFG. In this system we have a manufacturing line where parts flow along the line to processing stages where some work is done on the parts before they are sent to the next stage. We present a model for one of the stages. The operation of this stage (say **Stage X**) is defined as follows.

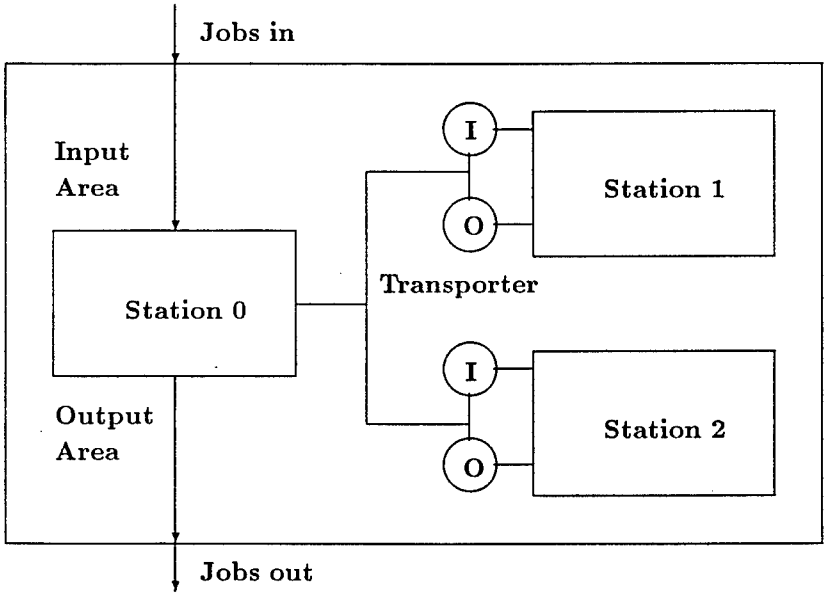


Figure 7: Overview of Stage X

Parts arrive at a large capacity input area (station 0). Parts are then moved by a single transporter to one of two identical workstations (stations 1 and 2) for processing. After parts finish processing at a

workstation they are moved by the transporter back to a large capacity output area at station 0. From the output area, processed parts leave the this stage. The transporter which moves parts between stations has a capacity of one part. An overview of the **Stage X** system is given in Figure 7.

Workstations 1 and 2 each have a finite input buffer, a finite output buffer, and a machine which processes parts. A workstation retrieves a part from its input buffer, does some processing on the part, and then returns the processed part to its output buffer.

The transporter moves parts from the input area at station 0 to input buffers of the workstations and moves processed parts from output buffers of the workstations to the output area at station 0.

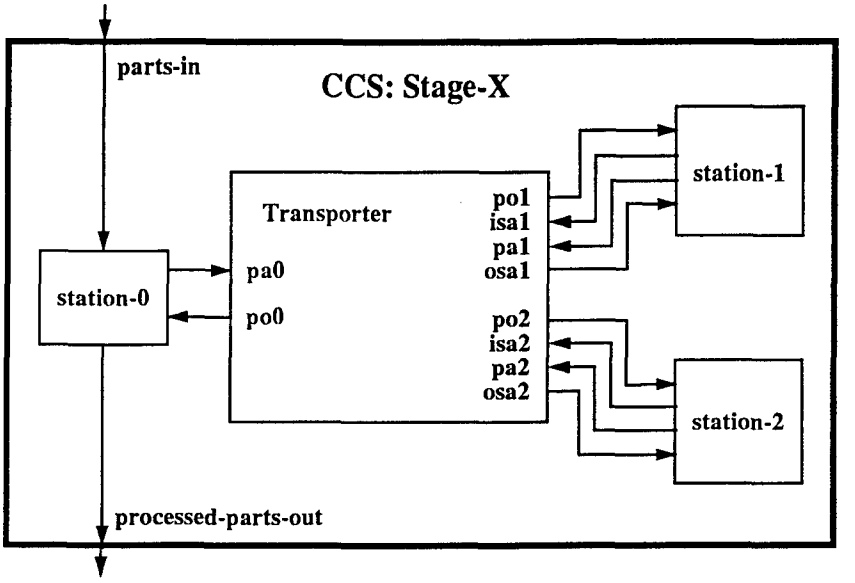


Figure 8: Internal View of Stage X Component

We model a system by recursively decomposing higher level components into lower level components which become the AC's of the model. The CCS for coupled component **Stage X** (root node of the HIG tree) is shown in Figure 8. We decompose **Stage X** into four components: an input station (**station 0**), two workstations (**station 1** and **station 2**), and a **transporter**. We model the **transporter** as an AC and show a partial specification using an HCFG's for the **transporter** AC.

The ports of the transporter AC are also shown in Figure 8. The ports and their uses in the model are described below.

Ports **pa0** and **po0** are used to communicate with the **Station 0** as follows:

pa0 – Messages arriving on this port indicate that a part has arrived at the input area at **station 0** for the transporter to take to a workstation. (part available at station 0)

po0 – The **transporter** sends a message to this port when it unloads a processed part at **station 0** .

(part out to station 0)

Ports **po1**, **isa1**, **pa1**, and **osa1** are used by the transporter to communicate with the first workstation station 1 as follows

po1 – The **transporter** sends a message to this port to tell the workstation that a part has been placed in its input buffer. (part out to station **1**)

isa1 – A message arrival on this port indicates that the workstation removed a part from its input buffer. (input space available at station **1**)

pa1 – A message arrival on this port indicates that the workstation placed a processed part in its output buffer. (part available at station **1**)

osa1 – The **transporter** sends a message to this port to tell the workstation that a processed part was removed from its output buffer. (output space available at station **1**)

Ports **po2**, **isa2**, **pa2**, and **osa2** are used to communicate with **station 2** in the same manner as those for **station 1** above.

In the HCFG we specify behavior using a recursive top-down decomposition. Behavior is represented as control states, macro control states(MCS's), and the transitions (including state transformations) between those states. Although it is the recursive decomposition of the behavior of a component that generates the HCFG tree, we begin by showing the partial HCFG tree that results from the partial behavior decomposition (Figure 9).

The MaCSS for the root node of the HCFG tree (labeled **Top**) is shown in Figure 10. The top level MaCSS partitions the transporter's behavior into nine MCS's. These nine MCS's include the behavior when the transporter is located at each of the three stations and the behavior while the transporter is traveling between stations.

The behavior of the transporter while traveling between stations is simple and therefore we model the internals of the travel MCS's without further decomposition (i.e, only (non-macro) control states are contained within them and thus the travel MCS's are leaf nodes in the HCFG tree). The behavior of the transporter while traveling from one station to another station is a simple delay which indicates the travel time required to move from one station to other.

We define a simple MCS **Delay** as shown in Figure 11. A delay MCS has a single input pin, a single output pin, and a single control state. The edge from the control state to the output pin is a time delay edge which specifies how long after arrival at the control state the point of control should remain at the control state. This **Delay** MCS can then be reused as needed in the HCFG by simply supplying a parameter so that the **Delay** MCS knows how long the delay should be. Thus we can define the travel MCS's as

travel-ij = **Delay**(travel-time-from-station-i-to-station-j)

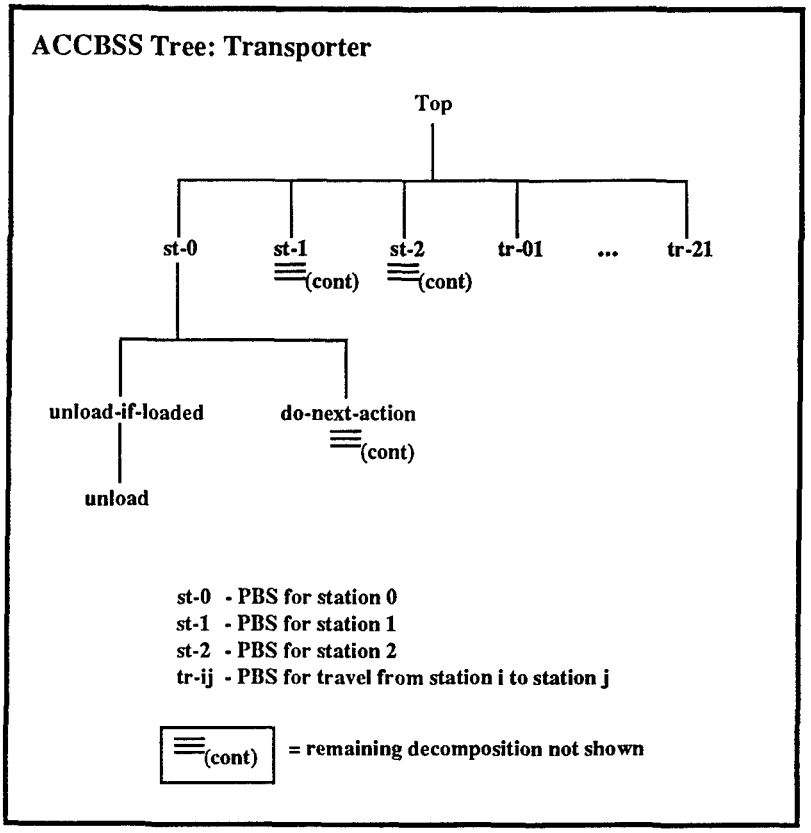


Figure 9: A Partial HCFG tree the Transporter AC

We next model the behavior of the transporter when it arrives at station 0. When the transporter arrives at station 0 it first unloads the processed part if it is loaded and then decides what to do next. This behavior is shown in the MaCSS for **Station 0** in Figure 12. This MCS has two input pins, two output pins, and two MCS's. One MCS **unload-if-loaded** specifies the behavior of the transporter when it first arrives at station 0, and the other MCS **do-next-action** specifies what the transporter should do next. We do not present the decomposition of the **do-next-action** MCS in this example.

In the **unload-if-loaded** MCS the transporter has to unload a part if it is currently loaded. We model unloading a part as a time delay. With some forethought we see that the MaCSS structure for **unload-if-loaded** appears to be a structure that can be reused. So we define a MCS **Delay-if-true** as shown in Figure 13. The **Delay-if-true** MaCSS needs two parameters

- how long to delay, and
- on what condition to delay

If the condition is *True*, the point of control MaCSS passes to the **Delay** MaCSS. The **Delay** MaCSS knows how long to delay because the **Delay-if-true** MaCSS passes the delay length information that it

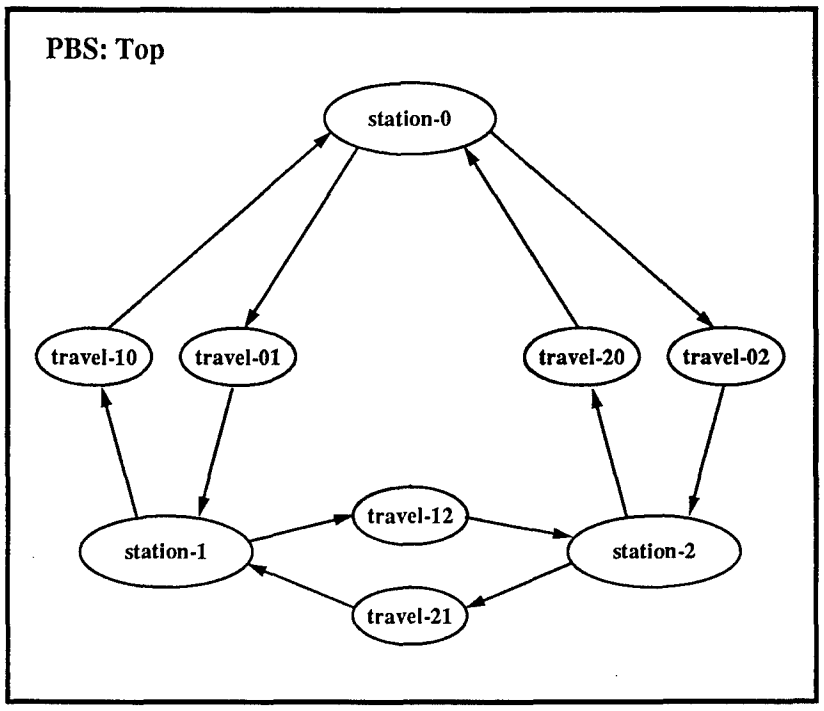


Figure 10: Top level MaCSS for the Transporter AC

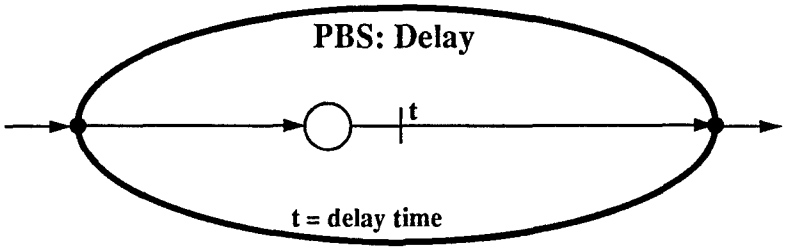


Figure 11: Internal View of a Delay MaCSS

receives to the **Delay** MaCSS. If the condition is *False*, the **Delay-if-true** MaCSS bypasses the **Delay** MaCSS and passes control out of the MaCSS via the output pin without any time delay. Thus we can specify the **unload-if-loaded** MaCSS as

unload-if-loaded = **Delay-if-true**(unload-time, loaded?)

where *unload-time* is the time required to unload a part from the transporter which is passed to the **Delay** MCS and *loaded?* is an expression which is used as the predicate **p** in Figure 13 that evaluates to *True* if the transporter is loaded with a part, or *False* otherwise.

Although we do not show the HCFG subtree (and associated MaCSS's) for the transporter behavior while in the workstation MCS's (**station 1** and **station 2**), since the two workstation MCS's are identical

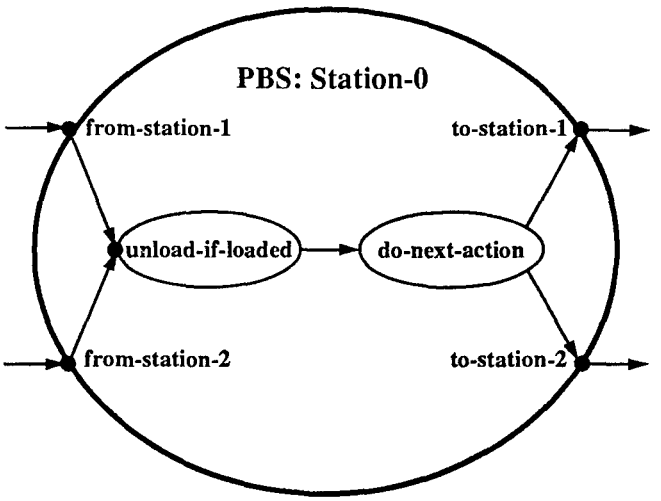


Figure 12: Internal View of the MaCSS for **Station 0**

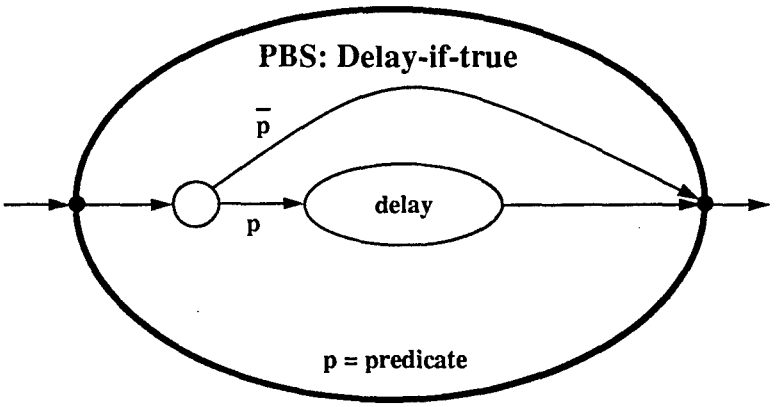


Figure 13: **Delay-if-True** MaCSS

(except for some parameters) we could generate and use a single workstation MCS for both workstation MCS's. As long as the MaCSS's for the workstations are designed with reuse in mind, the entire subtrees of the HCFG tree could be reused simply by supplying a different set of parameters. This indicates that there are two broad categories of reusable MaCSS structures. One category is generic MaCSS's such as **Delay** and **Delay-if-true** which can be used across a wide range of models. The other category is application specific MaCSS's such as the MaCSS for **station 1** and **station 2** where we can reuse an entire subtree of the HCFG tree.

6 Mapping HCFG Models to CFG Models

According to Zeigler[Zei90], a hierarchical model can be executed either using a hierarchical simulator, or by first flattening the model to form an equivalent nonhierarchical model. In this section we present algorithms to map an arbitrary HCFG Model into an equivalent CFG Model. Once we have an equivalent CFG Model we can execute the model using existing CFG Model algorithms. Since the HIG and the HCFG structures differ significantly, we use two distinct mapping algorithms. One algorithm maps an HCFG into a CFG and the other reduces a HIG to an IG.

6.1 Mapping an HCFG to a CFG

To map an HCFG into a CFG, we need to reduce the HCFG tree to a depth of zero (only the root node remains). We do this by repeatedly selecting a MCS contained within the root MCS and incorporating the internals of that MCS into the root MCS, thereby eliminating the child MCS by replacing it with its internals, while preserving the correct mappings and uniqueness of names. The algorithm for mapping an HCFG to a CFG is (using the notation of section 5.4):

$G_j \leftarrow$ root node of the HCFG tree

while ($\exists$ a child MCS $G_i \in \mathcal{G}_j$) {

- select a child MCS $G_i \in \mathcal{G}_j$
- give G_i a unique identifier prefix label $\mathcal{L}_i$
- make the encapsulation boundary around G_i "transparent" (G_i's boundary still exists, but the internals of G_i become visible to G_j)
- for each local variable defined in G_i, prefix that variable with $\mathcal{L}_i$ and add that variable to the local variable set of G_j (Note that because of the unique prefix $\mathcal{L}_i$ that we cannot have any name collisions in the G_j name scope.)
- for each reference to a local variable in G_i replace that reference with a reference to the variable in name scope G_j that was formed by prefixing the variable's identifier in scope G_i with $\mathcal{L}_i$ (E.g., if an edge condition in G_i refers to a variable X, replace that reference to X with a reference to the variable $\mathcal{L}_i X$ in G_j.)
- for each reference to a handle in G_i, replace that reference to a handle with a reference to the handle that was passed to G_i by G_j during model construction (E.g., if G_j passed handles $\langle A, B, C \rangle$ to G_i and G_i refers to handles $\langle X, Y, Z \rangle$ then all references in G_i to X, Y, and Z are replaced with references to A, B, and C, respectively.)
- remove all local variables in G_i (At this point all references to the original local variables in G_i have been replaced with references in G_j.)

- for each edge e_k in G_j that terminates on an input pin p_i of G_i, change the termination point of edge e_k to the termination point of the edge that originates from the external input (the internal view of) pin p_i (This new termination point is inside G_i so edge e_k now crosses G_i's encapsulation boundary.)
- for each edge e_k in G_i that terminates on an external output pin p_o of G_i, change the termination point of edge e_k to the termination point of the edge that originates from pin p_o (This new termination point is outside G_i.)
- remove the encapsulation boundary of G_i, all pins of G_i, and all edges which originate or terminate on pins of G_i (All of the internals of the original G_i have been incorporated into G_j.)

} // end while

At the completion of this algorithm, the HCFG tree consists of the single root node represented by a single MCS which contains all the control states of the original HCFG, thus the MaCSS for the root node after the flattening represents a CFG.

The HCFG facilitates in the reuse of model elements at a sub-AC level. In an HCFG we can take encapsulated partial behavior specifications in the form of MaCSS's from a MaCSS library for reuse. In an HCFG Model we generally do a recursive top-down decomposition of AC behavior until we reach a point where all remaining MCS's are specified either by reusing existing MaCSS's or by developing a custom MaCSS which contains only control states.

6.2 Mapping a HIG to an IG

All causal behavior specification in an HCFG Model is internal to the AC's of that model. Each AC communicates with the outside world solely by sending messages to its output ports and receiving messages from its input ports. (Technically these ports are the *external* output ports and *external* input ports of the component since we are referring to the internal view of the AC, but in the case of AC's, there is no ambiguity as to which ports we mean, so we can drop the "external" part from the name.) In an HCFG Model all messages originate inside AC's and are sent to output ports of the AC's. All messages sent to output ports of AC's are routed as specified by the CISS and arrive unmodified at input ports of AC's.

All HCFG Models which have the same set of AC's and in which the same AC output ports are *eventually* mapped to the same AC input ports are behaviorally equivalent. Thus multiple HCFG Models can map to the same CFG Model.

Any HCFG Model CISS tree can be algorithmically reduced to depth of one while maintaining behavioral equivalence with the original HCFG model. In this reduced CISS tree (of depth one) we have a single coupled component at the top of the hierarchy and the subcomponents of this coupled component are the original atomic components (leaf nodes) of the original CISS tree.

This depth reduction can be accomplished by iteratively removing coupled subcomponents from the coupled component at the top of the CISS tree and replacing them with their internal representations (components and coupling) until the only remaining subcomponents are the atomic components (leaf nodes) of the original CISS. At this point the HCFG Model CISS is a rooted tree with a depth of one, all coupling is internal coupling, and all subcomponents are atomic components. This HCFG Model CISS tree with a depth of one is behaviorally equivalent to the original HCFG Model. An algorithm to reduce the depth of an HCFG Model CISS to one is given below. In the algorithm β^E refers to the external name of a port and β^I refers to the internal name of the same port. Notations in that algorithm below are those from the general CISS definition given in section 3.

$M_j \leftarrow$ root node(CCS) of CISS tree

while ($\exists M_i \in \mathcal{M}_j \ni M_i$ is a coupled component) {

- select $M_i \in \mathcal{M}_j \ni M_i$ is a coupled component
- $\mathcal{M}'_j \leftarrow \mathcal{M}_j \bigcup \mathcal{M}_i - \{M_i\}$
- define $\Omega'_j \leftarrow \Omega_j \bigcup \Omega_i$
- define $\Upsilon'_j \leftarrow \Upsilon_j \bigcup \Upsilon_i$
- define $\Gamma'_j : \Omega'_j \mapsto \Upsilon'_j \ni (\alpha, \beta) \in \Gamma'_j$ iff $\alpha \in \Omega'_j$, $\beta \in \Upsilon'_j$ and either $(\alpha, \beta) \in \Gamma_j$ or $(\alpha, \beta) \in \Gamma_i$
- $\forall \beta^E \in I_i$ do:
  - remove (α, β^E) from Γ'_j
  - remove β^E from Υ'_j
  - remove (β^I, γ) from Γ'_j
  - remove β^I from Ω'_j
  - add (α, γ) to Γ'_j
- $\forall \beta^E \in O_i$ do:
  - remove (β^E, γ) from Γ'_j
  - remove β^E from Ω'_j
  - remove (α, β^I) from Γ'_j
  - remove β^I from Υ'_j
  - add (α, γ) to Γ'_j
- $\mathcal{M}_j \leftarrow \mathcal{M}'_j$
- $\Omega_j \leftarrow \Omega'_j$
- $\Upsilon_j \leftarrow \Upsilon'_j$
- $\Gamma_j \leftarrow \Gamma'_j$

}

In an HCFG Model CISS tree with a depth of one, the CCS of the root node of the CISS tree specifies identical information to that of a CFG Model IG. Thus an HCFG Model CISS tree with a depth of one is equivalent to a CFG Model IG and we can use the CCS structure $M_j^C = (\mathcal{M}_j, \Gamma_j)$ to represent a CFG

Model IG. Thus, our algorithm to reduce the depth of an HCFG Model CISS tree to one also maps an HCFG Model CISS to an equivalent CFG Model IG.

The hierarchical CISS also facilitates in the reuse of model components. In a CFG Model we can pull encapsulated atomic components from a component library for reuse in a new model. In an HCFG Model we can do a bottom-up composition of lower level components to form encapsulated coupled components which can then be used in the construction of new models. This reuse of coupled components in addition to the reuse of atomic components gives us reuse capability at a "higher" level because coupled components encapsulate one or more atomic components.

We can now map any HCFG Model into an equivalent CFG Model. We next present a relatively detailed description of the two sequential algorithms from Cota and Sargent [CS90c] for the execution of CFG Models.

7 Summary

We have developed a general paradigm which uses two types of specification structures for specification of models based on encapsulated AC's which interact solely via message passing. We have shown how this paradigm can support hierarchical model specification and reuse of model specification elements. We then developed the Hierarchical Control Flow Graph Model paradigm which supports two types of hierarchical extensions of Cota and Sargent's Control Flow Graph Models. The HCFG Model paradigm (and the CFG Model paradigm) are special cases of our general model specification paradigm.

We have presented a systems theoretic formalism and an abstract simulator for HCFG Models. We have also specified a "flattening" algorithm to map an HCFG Model into the equivalent CFG Model. This allows a modeler to develop hierarchical models which can then, after flattening, be executed using existing CFG Model algorithms. The primary advantage of CFG Models is that they can be executed on either a sequential processor system or a parallel/distributed multi-processor system without the modeler adding addition information to the model, as is generally required in most existing parallel DES paradigms. Finally, we have presented two of the sequential algorithms for execution of CFG Models.

References

[Alp91] Alphatech, Burlington, MA. *Model 1 User's Manual*, October 1991.

[Bal88] O. Balci. The implementation of four conceptual frameworks for simulation modeling in high level languages. In M.Abrams, P.Haigh, and J.Comfort, editors, *Proceedings of the 1988 Winter Simulation Conference*, pages 287–295, 1988.

[BCM87] R. L. Bagrodia, K. M. Chandy, and J. Misra. A message–based approach to discrete-event simulation. *IEEE Transactions on Software Engineering*, 13(6):654–665, June 1987.

[BL90] R. L. Bagrodia and W. Liao. Maisie: A language and optimizing environment for distributed simulation. In D. Nicol, editor, *Distributed Simulation*, pages 56–59, 1990. The Society for Computer Simulation.

[Boo91] G. Booch. *Object-Oriented Design.* Benjamin/Cummings, 1991.

[CGLM93] K.C. Chang, R.F. Gordon, P.G. Loewner, and E.A. MacNair. The research queueing package modeling environment (resqme). In G.W. Evans, M. Mollaghasemi, E.C. Russell, and W.E. Biles, editors, *Proceedings of the 1993 Winter Simulation Conference*, pages 294–302, 1993.

[CS89] B.A. Cota and R.G. Sargent. Automatic lookahead computation for conservative distributed simulation. CASE Center Technical Report 8916, Syracuse University, December 1989.

[CS90a] B.A. Cota and R.G. Sargent. Control flow graphs: A method of model representation for parallel discrete event simulation. CASE Center Technical Report 9026, Syracuse University, December 1990.

[CS90b] B.A. Cota and R.G. Sargent. A framework for automatic lookahead computation in conservative distributed simulations. In D. Nicol, editor, *Distributed Simulation*, pages 56–59, 1990. The Society for Computer Simulation.

[CS90c] B.A. Cota and R.G. Sargent. Simulation algorithms for control flow graphs. CASE Center Technical Report 9023, Syracuse University, November 1990.

[CS92] B.A. Cota and R.G. Sargent. A modification of the process interaction world view. *ACM Transactions on Modeling and Computer Simulation*, 2(2):109–129, April 1992.

[dLF93] D. de Champeaux, D. Lea, and P. Faure. *Object-Oriented System Development.* Addison-Wesley, 1993.

[FS93] D.G. Fritz and R.G. Sargent. Hierarchical control flow graphs. CASE Center Technical Report 94xx, Syracuse University, December 1993.

[Fuj90] R.M. Fujimoto. Parallel discrete event simulation. *Communications of the ACM*, 33(10):30–53, October 1990.

[GB89] G.Lomow and D. Baezner. A tutorial introduction to object-oriented simulation and Sim++. In E.A. MacNair, K.J. Musselman, and P. Heidelberger, editors, *Proceedings of the 1989 Winter Simulation Conference*, pages 140–146, 1989.

[LL89] Y. Lin and E.D. Lazowska. Exploiting lookahead in distributed/parallel simulation. Technical Report 89-10-06, Department of Computer Science and Engineering, University of Washington, October 1989.

[Nic88] D.M. Nicol. High performance parallelized discrete event simulation of stochastic queueing networks. In M.Abrams, P.Haigh, and J.Comfort, editors, *Proceedings of the 1988 Winter Simulation Conference*, pages 306–314, 1988.

[Nic90] D.M Nicol. A "conservative" approach to parallelizing the sharks world simulation. In O. Balci, R.P. Sadowski, and R.E. Nance, editors, *Proceedings of the 1990 Winter Simulation Conference*, pages 186–190, 1990.

[Sar88] R.G. Sargent. Event graph modelling for simulation with an application to flexible manufacturing systems. *Management Science*, 34(10):1231–1250, October 1988.

[Sar92] R.G. Sargent. Requirements of a modeling paradigm (part of panel: Discrete event simulation modeling: Directions for the 90s). In J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, editors, *Proceedings of the 1992 Winter Simulation Conference*, pages 780–781, 1992.

[Str91] B. Stroustrup. *The C++ Programming Language.* Addison–Wesley, second edition, 1991.

[Wag91] D.B. Wagner. Algorithmic optimizations of conservative parallel simulation. In V.Madisetti, D.Nicol, and R.Fujimoto, editors, *Advances in Parallel and Distributed Simulation*, pages 25–32, 1991. The Society for Computer Simulation.

[Wei71] G.M. Weinberg. *The Psychology of Computer Programming.* Van Nostrand Reinhold, New York, 1971.

[Zei84a] B.P. Zeigler. *Multifacetted Modelling and Discrete Event Simulation.* Academic Press, 1984.

[Zei84b] B.P. Zeigler. System-theoretic representation of simulation models. *IIE Transactions*, 16(1):19–34, March 1984.

[Zei90] B.P. Zeigler. *Object-Oriented Simulation with Hierarchical, Modular Models.* Academic Press, 1990.

Metamodelling with Logistic Regression Techniques : Applications for TAC BRAWLER

by

Hunter Nichols

and

Dr. Jeffrey D. Tew

Department of Industrial and Systems Engineering
Virginia Polytechnic Institute and State University
Blacksburg, Viriginia 24061

Metamodelling with Logistic Regression Techniques :
Applications for TAC BRAWLER

by
Hunter Nichols
and
Dr. Jeffrey D. Tew
(ABSTRACT)

Logistic regression metamodelling is used in the analysis of simulation models in which the performance metric is binary (i.e. success vs failure) or polytomous (multiple discrete increments). By providing the user with a mathematical relationship between the model's performance and the values of input variables, its use offers the practitioner insight into the effect of input variables on the probability of success and into the interrelationship between the variables. This insight facilitates the task of system assessment and decision generation.

This document provides a brief outline of the metamodelling process and includes a summary of the model underlying logistic regression. In addition, an explanation of the need for experimental design and description of recent research on the optimal design of experiments for the analysis of binary performance metrics is included. This is followed by an example of logistic metamodelling using TAC BRAWLER data. Although the document is neither comprehensive or conclusive, it should provide practitioners a practical basis from which to implement the use of logistic regression techniques in metamodelling with TAC BRAWLER.

Table of Contents

1. Introduction

One of the tools available for the analysis of air combat systems and doctrines is a tactical simulation model called TAC BRAWLER. This software is designed to offer analysts the ability to simulate few-on-few aerial engagements, with a great degree of flexibility in the external stores configuration (missiles, countermeasures, etc), aircraft performance envelopes, and tactical doctrines. While this high degree of dimensionality enhances the value of TAC BRAWLER with respect to the range of applicable scenarios, this advantage is tempered by the associated difficulty in the design of experiments and the interpretation of output. In particular, the dimensionality of the design space makes it difficult to gain insight into the interrelationships between the input factors, or to understand the effect of the factors on the performance metric, which are two primary interests in the analysis of the output. Metamodelling addresses these concerns through the use of concepts in experimental design and parametric modelling which facilitate analysis of the general behavior of the system.

In Section 2, we briefly cover the notation and concepts underlying least-squares and logistic metamodelling techniques and differentiate between their applications. This discusssion also addressses the topics of planned experiments and optimal experimental design. In Section 3, we present an example of the application of logistic metamodelling, and in Section 4, we provide some conclusions and recommendations.

2. Metamodels

Within the field of simulation, the term metamodelling is used to describe the use of regression techniques in the analysis of output from simulation models. In particular, metamodelling is concerned with the estimation of a mathematical relation between input factors and the system's performance. In actuality this process could be described as "modelling a model", as in this case we are using statistical modelling techniques to analyze the output from a simulation model. This usually takes the form of an empirical process where we design an experiment to yield information from which inferences can be drawn about the system's performance over a range of input values.

The most common metamodelling approach corresponds to least squares regression. This class of models is characterized by the fact that the parameters are linear, i.e. the parameters raised to the first power. This is commonly represented mathematically by

$$\mathbf{Y} = \mathbf{X}\beta + \varepsilon. \tag{1}$$

There are three assumptions on which least-squares regression is based. The first assumption is that the error term ε_i is a random variable with mean μ zero and variance σ^2, (i.e. $E(\varepsilon_i)=\mu=0$ and $V(\varepsilon_i)=\sigma^2$). It is also assumed that given i = j, ε_i and ε_i are uncorrelated, or $COV(\varepsilon_i, \varepsilon_j)=0$. The significance of these assumptions is that $E(Y_i)=\beta_0+\beta_1\mathbf{X}$ and $V(Y_i)=\sigma^2$, and that Y_i and Y_j are uncorrelated. The third assumption is that ε_i is normally distributed, which together with the first assumption implies that $\varepsilon_i \sim N(0,\sigma^2)$. This implies that ε_i and ε_j are independent, and is essential in the construction of confidence intervals for estimated responses.

It should be noted that although the term regression is generally assumed to refer to least-squares linear modelling, it collectively encompasses a wide array of parametric modelling techniques. These include not only least-squares linear models, but also nonlinear models, "alternative" regression techniques (those not based on least-squares), and logistic modelling. To date, the majority of literature on metamodelling has focused on least-squares techniques for modelling systems where the performance is characterized as real and continuous. However, in situations where the performance can be characterized as binary or dichotomous (success vs. failure) or polytomous (a fixed number of discrete incremental levels), logistic regression is a more appropriate approach.

Although the analytic techniques used in logistic regression are similar to those used in least-squares regression, both the underlying assumptions and choice of parametric model differ for the two methods. In the case of least-squares methods, the estimated response is a conditional mean which can vary over the entire range of real numbers, while in logistic regression it varies (inclusively) between zero and one. This constraint is reflected in a cumulative function which is a symmetic signoid curve between zero and one.

Let $\pi(\mathbf{x})$ denote the conditional mean of Y given $\mathbf{x}$, $E(Y|\mathbf{x})$. The specific form of the regression model which corresponds with the use of the logistic distribution is

$$\pi(x) = \frac{e^{\beta_0+\beta x}}{1+e^{\beta_0+\beta x}} \tag{2}$$

A fundamental difference between the linear and logistic models is the conditional distribution of the response variable. In the linear model it is assumed that a response can be expressed as

$$y = E(Y|x) + \varepsilon \tag{3}$$

and that the error term ε is normally distributed. with a mean of zero and a variance which is constant across all levels of the independent variables. This implies that the conditional distribution of the response Y, given x, is also normally distributed, but with mean $E(Y|x)$ and a constant variance.

In contrast, based on the term $\pi(\mathbf{x})$ defined above, a dichotomous response can be expressed as

$$y = \pi(x) + \varepsilon . \tag{4}$$

The error term ε can assume two values, depending on the value of y. If y=0 then $\varepsilon=-\pi(x)$ with a probability of $1-\pi(x)$, and if y=1 then $\varepsilon=1-\pi(x)$ with a probability of $\pi(x)$. This implies that the error term has a binomial distribution, with a mean 0 and variance $\pi(x)(1-\pi(x))$. Thus the response is bounded between 0 and 1, inclusively, with a binomial distribution with mean $\pi(x)$. Given that $\pi(x)$ is nonlinear, the error term is not normally distributed, and the assumption of a constant variance across the range of the independent variables is incorrect, it follows that least-squares regression is inappropriate.

2.1 Logistic Metamodel Estimation

The estimation of a logistic metamodel is based on the method of maximimum likelihood estimators. To illustrate this process consider a set of data in which we have n responses (yi, i=1,2,...,n) and a set of factor levels (x_{ij}, i=1,2..,n and j=1,2,..p) given by

$$\begin{array}{cccccc} y_1 & x_{11} & x_{12} & \cdots & x_{1p} \\ y_2 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & & \vdots \\ y_n & x_{n1} & x_{n2} & \cdots & x_{np} \end{array} \tag{5}$$

Suppose we postulate the following logistic metamodel

$$y_i = \frac{e^{\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}}}{1 + e^{\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}}} + \varepsilon = \pi(x_i) + \varepsilon. \quad (6)$$

This can also be expressed in vector notation as

$$y = \frac{e^{X\beta}}{1 + e^{X\beta}} + \varepsilon = \pi(x) + \varepsilon. \quad (7)$$

As in the case of the least-squares metamodel, the parameters are estimated using the maximum likelihood principle. This approach yields parameter estimates which maximize the probability of obtaining the observed set of data. By constructing a likelihood function which expresses the probability of the observed data as a function of the unknown parameters, and finding the parameter values which maximize this function, the analyst obtains the maximum likelihood estimators.

If the response is coded as zero or one (i.e. dichotomously) then $\pi(\mathbf{x_i})$ yields Prob$(1|\mathbf{x_i})$, the conditional probability that the response will equal one, given x_i. The conditional probability that the reponse equals zero is given by $1-\pi(x_i)$. It follows that in the case where an observed response y_i equals one, the contribution to the likelihood function is $\pi(x_i)$, and where an observed response equals zero the contribution is $1-\pi(x_i)$.

For a given set of factor levels and the associated response, this can expressed mathematically as

$$\zeta(x_i) = \pi(x_i)^{y_i}\left[1 - \pi(x_i)\right]^{1-y_i}. \quad (8)$$

Assuming that the responses are independent of one another, the likelihood function equals the product of these terms over the set of response values, as folows:

$$l(\beta) = \prod_{i=1}^{n} \zeta(x_i). \quad (9)$$

In order to simplify the task of maximizing this function, the log-likelihood is often used. This is

$$L(\beta) = \ln[l(\beta)] = \sum_{i=1}^{n} \{y_i \ln[\pi(x_i)] + (1 - y_i) \ln[1 - \pi(x_i)]\}. \quad (10)$$

In order to maximize L(β) we differentiate it with respect to β_0 and β_1 and set the expressions equal to zero. The resulting equations are

$$\sum_{i=1}^{n} [y_i - \pi(x_i)] = 0. \quad (11)$$

and

$$\sum_{i=1}^{n} x[y_i - \pi(x_i)] = 0. \quad (12)$$

Whereas solving for the parameters in a least-squares metamodel is fairly straightforward due to the linear properties of the parameters, solution of these equations requires a more sophisicated approach. The available methods most often used include iterative weighted least-squares, noniterative weighted least-squares, and discriminant analysis. Most major statistical software packages (such as SAS, BDMP, GLIM, and SPSS-X) use an iterative weighted least-squares algorithm for this purpose. (See Hosmer and Lemeshow, 1989, p.18)

2.2 Interpretation of the Parameters

In the modelling process a natural question which arises is the meaning of the parameters; what information do they provide? In contrast to least-squares modelling, the information in the estimates in a logistic metamodel is not immediately apparent. However, with the use of an appropriate link function the task of interpretation is simplified.

A link function provides a linear function between the response and independent variables. For the least-squares model this is the identity function (y=y) and for the logistic model the link function is the logit transformation, which is also known as the log-odds ratio. Defined in terms of $\pi(x)$ this transformation is

$$g(x) = \ln\left(\frac{\pi(x)}{1 - \pi(x)}\right) = \beta_0 + \beta x. \quad (13)$$

In the least squares models the parameter can be interpreted to be the difference in value of the response at x and x+1. Following this approach, the parameter in a logistic metamodel can be viewed as the difference in the logit between x and x+1 (i.e. $\beta_i=g(x_i+1)-g(x_i)$). To understand what this means requires an understanding of what the difference between two logits represents (Hosmer and Lemshow, 1989, p.39).

The odds-ratio is defined as the ratio of odds for success to the odds for failure, and is given by the equation

$$\psi = \frac{\pi(success)/[1-\pi(success)]}{\pi(failure)/[1-\pi(failure)]}. \tag{14}$$

The log-odds ratios is the log of this term, and is given by

$$\text{Log}(\psi) = \text{Log}\left[\frac{\pi(\text{success})/[1-\pi(\text{success})]}{\pi(\text{failure})/[1-\pi(\text{failure})]}\right] = g(\text{success}) - g(\text{failure}). \tag{15}$$

Noting that this is the difference between two logits, it follows that difference between two logits equals the log-odds ratio. In the case of the continuous variable in a univariate model, let c equal the change in the independent variable. Then $g(x+c)-g(x)=c\beta_1$ and we obtain the associated odds ratio by exponentiating this term (i.e. $e^{c\beta 1}$). This can be interpreted to mean that success is $e^{c\beta 1}$ more likely at $x+c$.

2.3 Testing for Significance

Once the parameters have been estimated the next task is to assess the significance of the independent variables with respect to the response. In particular, this addresses the question "Does a model including a variable tell us more about the response than one which excludes it?". It should be noted that this is a relative question, where we are interested in comparing the accuracy of two estimates with respect to the presence of the variable in question. This should not be confused with goodness-of-fit testing where we assess whether the predicted values are an adequate representation of the observed response values.

To determine the significance of a variable in a logistic metamodel, we compare the observed response to the predicted response in models with and without the variable. Comparison of observed to

predicted response is based on the log likelihood function L(β). Noting that an observed response is actually a predicted response from a saturated model (i.e. one where there as many parameters as data points), the likelihood ratio is defined as the likelihood of the current model divided by the likelihood of the saturated model. The deviance, denoted as D, is given by

$$D = -2\ln[\text{likelihood ratio}] = -2\sum_{i=1}^{n}\left[y_i \ln\left(\frac{p_i}{y_i}\right) + (1-y)\ln\left(\frac{1-p_i}{1-y_i}\right)\right]. \tag{16}$$

This term is analogous to the residual sum of squares in a linear regression analysis of variance. (See Hosmer and Lemeshow, 1989, p.14). Multiplying the log of the likelihood ratio by -2 yields a quantity which is distributed under the χ^2 distribution., thereby facilitating formal hypothesis testing of the significance of the variables. To determine the significance of an independent variable we compare the value of D with and without the variable in the equation. The difference is denoted as G, and is given by

$$G = D(\text{Model with variable}) - D(\text{Model without variable}). \tag{17}$$

Noting the definition of D above, this term can be expressed as

$$G = -2\ln\left[\frac{\text{likelihood with variable}}{\text{likelihood without variable}}\right] \tag{18}$$

Under the null hypothesis that the parameter associated with the independent variable of interest equals zero, this test statistic (G) is tested against the chi-square distribution with p degrees of freedom.

2.4 Model Adequacy and the Concept of Pure Error

In least-squares metamodels, model adequacy is assess by comparing the lack of fit to the pure error (See Myers, 1991, p.72). The residual sum of squares SS_E can be partitioned into two parts, SS_{LOF} and SS_{PE}. The lack of fit component is the weighted sum of the squared deviations between the estimated response at each **x** level and the mean observed response at each **x** level. The pure error is the corrected sum of squares (deviation between observed responses and the mean observed response at each x level) pooled over all the values of x. The statistical test for lack of fit is an F-test,

$$F = \frac{SS_{LOF}/(m-2)}{SS_{PE}/(n-m)} = \frac{MS_{LOF}}{MS_{PE}}. \tag{19}$$

A significant test statistic is interpreted to mean that the original hypothesized model is inadequate, and that a new model must be developed. Conversely, an insignificant test statistic results in a failure to reject the original model, and the mean squares for lack of fit and pure error are combined to estimate the variance.

In the case of logistic regression, we can compare the unexplained variation (SS_E-SS_{PE}) with an estimate of the true variance based on the pure error (SS_{PE}). It should be noted however that while this exercise will yield insight into the adeqaucy of the model, the statistical test is not valid in any formal sense (See Draper and Smith, p. 484) .

2.5 Metamodel Development Strategies

In the metamodelling process it is not an uncommon situation where there is no single fixed model, where our primary concern is the significance of each independent variable. Instead we begin the process with a basic parametric model and a pool of candidate independent variables, and carry out a series of steps which allow us to identify either the 'best' model, or more typically a set of candidate models which conform to our selection criteria.

In order to achieve this, a methodology or strategy is required which will first select the variables for the model and then allow the assessment of the model's adequacy with respect the individual variables and the overall fit of the model. (Hosmer and Lemeshow, 1989, p.82). In the process of selecting variables the goal is to develop a model which provides an adequate fit while avoiding the inclusion of superflous variables. Since including additional variables in the model results in increased standard errors and makes the model increasingly dependent on the empirical data, this approach to model development provides better numerical stability (Hosmer and Lemshow, 1989, p.83).

As a preliminary step in the selection process each independent variable in a univariate analysis. In the case of nominal and ordinal variables this is accomplished using contingency tables, relating the response variable (0,1) against the k levels of the independent variable. In cases where only a small number of specific values are used in a continuous independent variable, the contingency table approach is an option, but generally continuous variables are fitted to a univariate logistic model.

Suppose we have a 2 x k contingency table based on an independent variable and the response. The likelihood ratio chi-square test associated with the table, with k-1 degrees of freedom is equal to the likelihood ratio test for the significance of the coefficients in the univariate logistic model based on that

independent variable. The fact that the likelihood ratio chi-square test is asymptotically equivalent to Pearson's chi-square test allows us to use the latter in our analysis (Feinberg, 1977, p.40). This is convenient since most software packages provide Pearson's statistic in a contingency table analysis.

A condition which we must careful to detect prior to including a varaiable in a model is a contingency table containing a zero, or 'empty' cell. Including such a variable in a logistic model without addressing this conditio will result in an extremely unstable, and unreliable, result. A common approach to this problem is to collapse the level containing the empty cell into an adjacent level, or if the variable is nominal, finding a suitable category into which we can collapse the category containing the empty cell. Other approaches include eliminating the level or category entirely, or in the case of an ordinal variable, treating the independent variable as if it were a continuous variable (Hosmer and Lemeshow, 1989, p.84).

The recommended procedure for a continuous independent variable is to fit that variable in a univariate logistic model. This provides us with estimates of the associated coefficent and standard error, and allows us to perform the likelihood ratio test for the significance of the variable and obtain the Wald statistic (as shown in Section 2.2).

Once we have completed the univariate analyses, we use the results to screen the variables for further consideration. Those variables for which the p-value from the test for the significance of the parameter in the univariate analysis is less than 0.25 are considered as candiates for the multivariate model. Work in the areas of least-squares regression by Bendel and Afifi (1977) and in the area of logistic modelling by Mickey and Greenland (1989) support the use of this relatively high threshhold for selecting candidates for multivariate models (Hosmer and Lemeshow, 1989, p.86). Use of values such as 0.05 or 0.10, which are commonly used in hypothesis testing, often results in the exclusion of variables which contribute to the multivariate model.

Having identified the candidate variables, there are a number of approaches which can be used for variable selection. One method relies on the experience of the analyst to identify all the scientifically relevant variables, and includes all of these in the model. Unfortunately the sample size may not suppport such a model, resulting in a numerically unstable model with excessive standard errors. In such a case the recommended approach is to select a subset, based on the results of the univariate analyses.

Another approach is called stepwise model development where variables are either included or deleted from the model on the basis of statistical criteria. Typically this is implemented in one of two methods. The first is called forward selection with a test for backward elimination. The second is call

backward elimination with a test for forward selection (Hosmer and Lemeshow, 1989, p.87). Algorithms to implement these methods are available in most major statistical software.

One appealing aspect of the stepwise procedures is that in building the model in a sequential fashion it allows us to examine a set of models rather than a specific model. The examination of the models is an important step in determining not only which model best fits our needs, but also allows us to consider the variables included in each model and identify noise variables as opposed to those which are truly explanatory in nature.

Once the stepwise (or an altenative) procedure yields a model, our next step is to examine the Wald statistic for each variable. If the Wald statistic for a variable reflects an insignificant contribution to the model, then the variable should be dropped. The likelihood ratio for the new model is then compared to that of the original model, and the coefficients of the remaining variables are compared to those of the original model. If there is a large difference between the parameter estimates for a variable in the new model compared to the original model, this indicates that one or more of the variables which was deleted provided a needed adjustment for the variable (Hosmer and Lemshow, 1989, p.88) This iterative fitting procedure (subsequent to the stepwise procedure) is continued until a model is identified which has an acceptable likelihood ratio statistic, and those variables which are statistically unimportant have been eliminated. (It should be noted that although the stepwise procedure can be implemented through an algorithm, this fitting process is based on the experience and judgement of the analyst.)

The link function for the logistic model, which yields a linear function of the independent variables, is the logit transformation, which is also known as the log-odds ratio. Defined in terms of $\pi(\mathbf{x})$ this transformation is

$$g(x) = \ln\left(\frac{\pi(x)}{1-\pi(x)}\right) = \beta_0 + \beta x. \tag{20}$$

In the process of selecting our candidate variables we assumed that the logit is linear with respect to our independent variables. Once we have identified our candidate models, the next step is to check the validity of this assumption. Although it is common to assume that the logit, g(x), is linear with respect to the independent variables, this is not always the case. On occasion the logit can prove to be quadratic or some other nonlinear function, and in some cases is even binary (i.e. there exists a threshhold above and below which the logit assumes two distinct, constant respective values). As an initial procedure for checking this assumption, it is recommended that logit be plotted as a function of each of the independent

variables. For multivariate models this can be augmented by analyzing the logit using linear regression techniques to check for linearity.

A second approach is the Box-Tidwell transformation (Hosmer and Lemeshow,1989, p.90). In logistic modelling this is implemented by adding a $x\ log(x)$ to the model, and checking for significance. If the coefficient for this term proves to be significant, this is an indication that nonlinearity is present. Two shortcomings of this approach are that it doesn't alway detect minor departures from linearity and it doesn't provide any information on the nature of the nonlinearity. If nonlinearity is detected then the plotting and regression procedures mentioned above should be implemented.

Once the linearity of the logit is confirmed, or suitable terms have been added to address any nonlinearity, the next step in the model development is to assess the need for interaction terms. A need for an interaction term reflects a condition where the effect of one of the variable is not constant over the range of values of another variable. The test for this adding appropriate product (interaction) terms are added to the model and and a likelihood ratio test is used to test for their significance.

2.6 Model Assessment

After a final set of models is determined, they must be assessed in terms of comparative fit. Recall that -2 log(L) is distributed under the chi-square distirbution, and that G is also under this distribution. Taking the G statistic and associated degrees of freedom for each model, we determine the associated probabilities. For each model this is the probability of finding a statistic of this magnitude or larger as a result of random chance when the null hypothesis is true. The lower the statistic, and conversely, the higher the associated probability, the better the comparative fit. (Note the contrast between this and the criteria for testing for significance, where a low probability is desirable.)

An additional concern is the predictive performance of the models. A common use of logistic metamodelling is to develop a model for the purpose of predicting success under a range of operational conditions. Consider a 2 x 2 classification (contingency) table where the rows represent observed responses (success and failure, respectively) and the columns represent predicted responses (success and failure). A response is predicted to be a success if the probability

$$p = \frac{e^{X\beta}}{1+e^{X\beta}} \tag{21}$$

is found to exceed 0.5, otherwise it is predicted to be a failure. For the purpose of measuring the prediction perfomrance of a model, there are five metrics which are commonly used, and which are available in the output from most software. The primary metric is the percentage of correct predictions. A second metric, which is referred to as sensitivity represents the percentage of successes which are correctly predicted. The third metric, which is related to the second is called specificity and represents the percentage of failures which are correctly predicted. The fourth and fifth metrics are the the percentage of false positives (successes) and false negatives (failures), respectively.

In cases where prediction is an objective these metrics can be used to compare the performance of competing models. As indicated above, the primary metic is the overall percentage of correct predictions, but the other metrics can prove useful in cases where there is an inordinate risk or cost associated with false predictions of success (or failure).

In cases where the fit or prediction performance are inadequate or suspect, this leads us to examine the diagnostics for the models Pregibon (1981). These are broken into four categories. The first is the elements of the hat matrix diagonal. Those elements which are comparatively large, particularly those of an inordinate magnitude, are useful for identifying extreme points in the design space. These elements are calculated as

$$h_{jj} = n_j p_j (1-\hat{p}_j)(1, \mathbf{x}_j')\hat{\mathbf{V}}_\beta (1, \mathbf{x}_j')'. \tag{22}$$

where V_β is the covariance matrix for the parameters, xj is the set of values for the independent variables at observation j, and p_j represnts the probability associated with x_j.

A second set of diagnostics which are useful for identifying observations in our data which are poorly explained by the model are the Pearson residual and deviance residual. A Pearson residual is the element of Pearson's chi-square resulting from the inclusion of a particular observation. The Pearson residual for observation x_j is

$$\chi_j = \frac{(s_j - n_j\hat{p}_j)}{\sqrt{np_j(1-\hat{p}_j)}}, \tag{23}$$

where s_j is the number of success out of the n_j trials in associated with the j-th observation. Similarly a deviance residual is a component of the deviance, based on the log likelihood function. Defining the deviance as

$$D = -2\sum_{j=1}^{r}(m(p_j) - m(s_j / n_j)) \tag{24}$$

where $m(\lambda)=s_j log(\lambda)+(n_j-s_j)log(1-\lambda)$, we define the deviance residual $d_j$as follows:

$$d_j = \begin{cases} -\sqrt{2n_j \log(1-\hat{p}_j)} \quad where\ s_j = 0 \\ \sqrt{2(s_j \log\left(\frac{s_j}{n_j\hat{p}_j}\right)+(n_j-s_j)\log\left(\frac{n_j-s_j}{n_j(1-\hat{p}_j)}\right)} \\ where\ 0 < s_j < n_j\ and\ s_j / n_j > p_j \\ -\sqrt{2(s_j \log\left(\frac{s_j}{n_j\hat{p}_j}\right)+(n_j-s_j)\log\left(\frac{n_j-s_j}{n_j(1-\hat{p}_j)}\right)} \\ where\ 0 < s_j < n_j\ and\ s_j / n_j < p_j \\ \sqrt{-2n_j \log(\hat{p}_j)}\ where\ s_j = n_j \end{cases} \tag{25}$$

Where there is an inordinately large value of either residual, particularly in cases where it is greater than 2 or less then -2, the associated observation is not modelled well by the equation.

A third type of diagnostic is a measure which is useful for identifying observations which have a disproportionate effect on the maximum likelihood estimates (parameters). DFBETA is the standardized difference in the parameter estimate resulting from the removal of the respectiveobservation. For computational efficiency this is implemented using a one-step estimate of the change in parameter due to the deletion of each respective observation, rather than actually reestimating the parameters for each observation. Defining the on-step estimate as $\beta_j^1=\beta-\Delta\beta_j^1$, where

$$\Delta\beta_j^1 = \hat{\mathbf{V}}_\beta \mathbf{x}_j \frac{(s_j - n_j p_j)}{(1-h_{jj})} \tag{26}$$

we define DFBETA as

$$\text{DFBETA}_{ij} = \frac{\Delta_i \beta_j^1}{\hat{\sigma}(\beta_i)}, \tag{27}$$

where the denominator represents the standard error of β_i. This diagnostic is supplemented by two other measures C and CBAR, which are analogous to the Cook's distance concept used in least-squares regression (SAS/STAT, p.1094). These are computed as

$$C_j = \frac{\chi_j^2 h_{jj}}{(1-h_{jj})^2} \tag{28}$$

$$\overline{C}_j = \frac{\chi_j^2 h_{jj}}{(1-h_{jj})} \tag{29}$$

respectively.

Two other measures, DIFDEV and DIFCHISQ, are used to identify those observations which cause disproportionate disagreement between the observed data and the predictions. Respectively these represent the change in deviance and change in Pearson's chi-square statistic resulting from the absence of a given observation. These are computed as follows:

$$DIFDEV = \Delta_j D = d_j^2 + \overline{C}_j \tag{30}$$

$$DIFCHISQ = \Delta_j \chi^2 = \frac{\overline{C}_j}{h_{jj}} \tag{31}$$

2.7 Experimental Design

Recognizing that the purpose of an experiment is to gain insight into a system's performance, and that the experiment is not an end in itself, there are typically constraints on the available resources to devote to an experiment. Consequently the purposes of experimental design are two-fold. The first is to insure that the information from the experiment can be used in the inductive processes associated with the analysis of the system. Often taken for granted, this aspect of the experiment is critical in that failure to properly plan the experiment can results in output of little or no utility. This is due to situations where the effects of two or more input variables are confounded, or there exists an extraneous influence upon the systems which affects the response but is not accounted for in the design. This limits the analyst's ability to make any inferences with respect to the effect of the input variables on the response. (This particular

topic is beyond the scope of this paper, and for interested readers chapter 12 of Law and Kelton (1991), Hicks (1982), or any number of introductory experimental design texts would be recommended.)

The second purpose of experimental design is to increase the efficiency of an experiment; to determine a design which will yield the most information for a given expenditure of resources. Typically the resource constraint is manifested in a fixed number of trials (runs) due to limits on money and time. Three aspects of the design which are addressed at this stage are: what levels of the input variables should be covered, what combinations of these levels should be used (i.e. what design points), and how should the available trials be distributed among the design points (Adelbasit and Plackett, p.90). Two related issues which must be considered is the expected (or hypothesized) relationship of the response to the input variables, and the inherent variation in the response (Draper and Smith, p.52).

Experimental designs are gauged by two types of criteria. The primary criterion focuses on the quality of the information from the experiment. These tend to focus on either the variance-covariance matrix of β for the linear model, or on the information matrix for nonlinear cases such as the logistic model. Secondary criteria, which often conflict with the primary design criterion, reflect the need for the analyst to be able to validate the model and check the assumptions. Typically a design which reduces or minimizes the variance (or some related aspect of the problem) may prove to be inadequate with respect to the analyst's ability to check the validity of assumptions about the underlying distributions and variance. Thus the secondary criteria actual constrain the implementation of the primary criteria.

In the case of linear models there are a number of design criteria, but the most common is minimizing the β confidence space is the most common design criteria. Designs meeting this goal are called D-optimal, and are achieved by maximizing $|\mathbf{X'X}|$, which is inversely proportional to the square of the volume of the confidence space for β (Heise, 1993, p.3). A related criterion which is often used for linear models is to minimize the trace of $(\mathbf{X'X})^{-1}$, which implies that the sum of the variances of the parameter estimates is minimized, which is known as A-optimality. This approach ignores the covariance elements of $\mathbf{X'X}$, focusing instead on a measure of the overall variance of the estimates.

A third criterion, also based on $(\mathbf{X'X})^{-1}$, is to minimize the maximum eigenvalue of $(\mathbf{X'X})^{-1}$, which is called E-optimality. Large eigenvalues reflect the presence of large prediction variances, and in contrast to A-optimality uses the min-max concept rather than addressing an aggregate of the variances. This has been shown to be equivalent to minimizing the maximum prediction variance in the design space (Heise, 1993, p. 4).

Another criterion which addresses prediction variance is known as Q-optimality. This condition is achieved by minimizing the prediction variance integrated over the design space, and yield designs which have relatively uniform prediction variances over the design space (Heise, 1993, p. 5) An approach which contrasts with this is called G-optimality. A G-optimal design minimizes the maximum prediction variance over a given region (as opposed to the entire design space).

In the case of logistic metamodelling, there are analogous criteria to those used for linear models, but the nonlinear nature of the logistic model complicates the process. With respect to the maximum likelihood estimation of the parameters, the variance-covariance matrix for β is asymptotically the inverse of the Fisher information matrix, given by

$$I_{i,j} = -E\left[\frac{\partial^2}{\partial\beta_i \partial\beta_j}\log L(\beta)\right]. \tag{32}$$

In the case of linear models this equals $\sigma^2(\mathbf{X'X})^{-1}$, where the only unknown is the variance parameter, which is assumed to be constant. In the case of a nonlinear model, the information matrix is a function of the parameters which are being estimated, thus the variances of parameter estimates are dependent on the unknown parameters (Minkin, 1987, p.1098). Consequently much of the work on optimal designs for logistic regression have involved initial estimates and multi-stage (sequential) designs.

In logistic models, as in linear models, D-optimality is the most common criterion. This condition is achieved by maximizing the determinant of the information matrix, thus minimizing the volume of the likelihood based confidence space. In the case of a nonlinear model such as the logistic metamodelling, this is actually implemented by minimizing the determinant of the Fisher information matrix. Given that in effect this requires using a second order Taylor expansion of the log-likelihood function of the MLE, this is more appropriately known as local D-optimality (Minkin, p. 1098).

As mentioned above, the information matrix is dependent upon unknown parameters. Preferred approaches to this problem include initial estimates, multi-stage (sequential) methods, Bayesian methods, and constant information models. Prior information on initial estimates assumes that there exist estimates of the parameters from previous work or a pilot study. Much of the appeal of this method is tempered by the lack of robustness of the design criteria to errors in the initial estimate (Abdelbasit and Plackett, 1983, p. 90).

An extension of the initial estimates idea is the sequential design, where the parameter estimates evolve over a series of sequential experiments. In this approach the results of each stage are used as the initial

estimate for the next. Since in effect this is a repeated sampling procedure, one point of concern regarding this approach is the validity of confidence intervals, due to the lack of supporting asymptotic theory. Minkin (1987) has produced some empirical evidence to support the contention that there are few differences in the properties of the confidence intervals for fixed sample designs versus those of the a sequential design.

Bayesian methods require assigning prior distributions to the parameters and subsequently removing them from the analysis through the use of expectations. Chaloner and Larntz (1989) have produced Bayesian analogs to the D-optimal and A-optimal designs. One drawback to these approaches is that in cases where the uncertainty is high, the number of design points can be quite large (Heise, 1993, p.10).

Finally, it has been shown that under certain circumstances an experiment can be designed where the information is almost constant. It has been noted that this approach is limited to a rather narrow subclass of models and that very little supporting literature is available regarding this approach (Abdelbasit and Plackett, p.90).

To date nearly all of the experimental design work for logistic models has focused on the univariate model, due to the complexity of the design issue (Heise, 1993, p.7-8). Given the need for some form of initial estimate the focus of the work is two-fold; to identify an efficient design and to assess the robustness of the design with respect to errors in the estimates. From this work the a number of 2-point, 3-point, and 5-point designs have emerged, using both in single-stage and sequential approaches to the initial estimate problem.

The two more popular designs are the 2-point and 3-point designs, where there are N trials distributed among k uniformly spaced design points. Defining $\mu=-\beta_0/\beta_1$, the standard 2-point design places $N/2$ data points at $k=2$ design points located symmetrically about μ (assuming that N is even). The 3-point design also assumes symmetry about μ, but there are two different approaches. Earlier work suggests evenly dividing N over the three points, with the third point being μ, while later work simply emphasized symmetry, with at least $(N-1)/2$ points on each side of μ (Abdelbasit and Plackett, 1983, p.1099)

In a comparison of these two designs, the 3-point design emerged as a more robust design. Abdelbasit and Plackett (1983) mad the following observations with regard to these two designs:
(1) The 3-point design is more robust to the 2-point design in cases where the initial estimates are poor.
(2) In both designs, overestimating the parameter β_1 is a more serious problem than underestimating it. However, this discrepancy is mitigated as the error in the initial estimates decrease.

(3) In both designs the effects of poor initial estimates are exacerbated when the parameter β_1 is large, but again the effect is mitigated as the error in the initial estimates decrease.
(4) The effects of poor initial estimates cannot necessarily be compensated for by increasing the number of subjects at each level. However, this is more likely to happen if the number of design points, *k*, is increased.

One means of addressing the problem of poor initial estimates is to implement a multi-stage (sequential) design. For a two stage design there are two recommended approaches. The simplest method, suggested by Abdelbasit and Plackett (1983, p. 94-95) is to divide *N* by the number of stages and distribute these within each stages according to the design in that stage. In the two-stage designthis results in the data points being divided into four equal parts, with the x values determined as follows: $x_{1j}=(1.5434-\beta_0)/\beta_1$ and $x_{2j}=(-1.5434-\beta_0)/\beta_1$, where j=1,2, denoting the stage in the design. Using this approach, a tradeoff exists, where a higher stage design can improve the efficiency in cases where there are poor initial estimates, but where the additional stages are wasteful in instances where the estimates are relatively accurate (Abdelbasit and Plackett, 1983, p. 96).

A slightly more refined approach is to use information from the first stage to allocate the trials in the second stage (Minkin, 1987, p. 1100-1101). Let θ be the maximum likelihood estimate for the parameter, let $\theta_i=\beta_0+\beta_1 x_i$,and let $\omega_i=\omega(\theta_i)=\exp(\theta_i)/[1+\exp(\theta_i)]^2$. It can be shown that

$$\beta^2|J(\beta_0,\beta_1)|=\sum_{i,j}\omega_i\omega_j\theta_j^2-\left(\sum_i\omega_i\theta_i\right)^2 \qquad (33)$$

is maximized when $\theta_j=[\exp(\theta_i)+1/[\exp(\theta_i)-1]$, which yields $\theta_j=1.5434$ (Minkin, 1987, p. 1099). A suggested method for the second stage is to:
(1) Find the value of $\theta > 0$ the maximizes

$$2\omega_s^2\theta_s^2+\omega_s\theta_s^2(\omega_{1f}^{(s)}+\omega_{2f}^{(s)})+\omega_s[\omega_{1f}^{(s)}(\theta_{1f}^{(s)})^2+\omega_{2f}^{(s)}(\theta_{2f}^{(s)})^2]$$

(2) Compute

$$q=0.5-[\omega_{1f}^{(s)}(\theta_{1f}^{(s)})+\omega_{2f}^{(s)}(\theta_{2f}^{(s)})]/(4\omega_s\theta_s)$$

(3) Take $Nq/2$ data points at $x_{1s}=(\theta s-\beta_{0s})/\beta_{1s}$ and $N(1-q)/2$ at $x_{2s}=(-\theta s-\beta_{0s})/\beta_{1s}$.

One of the issues which arises withthese multi-stage designs is whether confidence spaces with repeated sampling properites can constructed onthe basis of the observed information (Minkin, 1987, p. 1101). On the basis of a simulation experiment there is little evidence to suggest that a multi-stage design will produce confidence intervals with lower coverage probabilities. However, it is noted that such intervals are systematically overestimated (Minkin, 1987, p. 1101-1102) .

More recently some work is emerging in the area of compromise designs. An example is a two-stage D-Q design in which the first stage is a 3-point D-optimal design with a subsequent stage in which a conditionally Q-optimal 2-point design is used. One aspect to note is that this design allows the levels (placement around μ) and allocation to be asymmetric, which is a more generalized approach than other designs have allowed. (Heise, 1993, p.10). (For further information the reader is referred to W. R. Myers, 1991).

3. TAC BRAWLER Example

In order to demonstrate the basic process of logistic metamodelling, we include an example where four factors are examined with respect to their influence on the probability of mission success. Although the authors lacked access to actual TAC BRAWLER software for the purpose of designing an experiment, the data used in this example is from an actual TAC BRAWLER experiment based on least-squares metamodelling (Zeimer and Tew, 1993). The scenario for this demonstration is a two on three engagement in which we have arbitrarily defined a successful mission as one in which the average kill ratio over ten trials is greater than one. The four factors which we examine are the number of radar guided missiles, the number of infrared missiles , the number of chaff cartidges, and the number of flares carried by each of the two blue aircraft. (A complete description and listing of the data is included in Appendix I.)

The experimental region for this experiment is defined with respect to the blue planes. The number of radar guided (RG) missiles ranges from one to 10, the number of infrared (IR) missiles ranges from one to nine, the number of chaff cartidges ranges from one to ten, and the number of flares ranges from zero to ten. This represents 9900 possible combinations of the four factors, but in the original experiment a 2 replication central composite design (CCD) with α=1was used (Zeimer and Tew, 1993). This resulted in 64 observations distirbuted over 25 distinct design points (this included additional replications for purposes of checking normality and variance assumptions).

As the first step in our analysis we subjected each of these four factors to a univariate analysis. Each of the four factors in this example are actually discrete incremental variables, but they can be analyzed as continuous variables in a logistic model.

Based on a univariate logistic model, testing for the significance of the parameter associated with the number of radar guided missiles (RG) yielded a Wald statistic of 9.3809 with 1 degree of freedom. Comparing this to a chi-square distribution with a single degree of freedom indicates that the probability of a test statistic of this magnitude or larger occurring due to random chance is 0.0022. This can be

interpreted to mean that the number of radar guided missiles is highly significant with respect to the probability of mission success. Consequently this variable is an excellent candidate for our pool of variables for subsequent development of a multivariate model.

Next we examine the number of infrared missiles as a factor in the probability of mission success. In this case the Wald statistic is only 1.5975, again with one degree of freedom. Comparing this relatively small statistic against the chi-square distribution indicates that the probability of finding a statistic of this magnitude or larger by random chance is 0.2063. Although this is not significant by normal standards used in univariate hypothesis testing, it falls under the suggested threshhold of 0.25 from Mickey and Greenland (1989), therefore we include it in our pool of candidate variables for the multivariate model.

Modelling the mission success against the number of chaff cartidges yields a Wald statistic of 1.5975 and like the infrared missiles the associated probability is 0.2063. Again, this isn't significant for a univariate model, but is sufficently low to include the variable in the pool of candidates.

Examining the univariate model of mission success as a function of the number of flares yields a Wald statistic of 2.4335. Comparing this to the chi-square statistic indicates that the probability of finding a statistic of this magnitude or higher is 0.1188, which is only marginally significant for a univariate test, but well under the threshhold for selecting candidates for the multivariate model.

At this stage all four of the original factors is included in our pool of candidates for the multiviate model. For this example we implement a forward stepwise algorithm (using PROC LOGISTIC in SAS) to develop the model.

Step 0 Enter the intercept

At this point the algorithm enters each variable in a univariate model and compares the chi-square statistic between each of the variables.

Table 1

Remaining Candidate Variables After Step 0

Variable	Chi-square	Prob > Chi-square
RG	10.0915	0.0015
IR	1.6146	0.2038
CHAFF	1.6146	0.2038
FLARES	2.4726	0.1158

The number of radar guided missiles has the lowest probability, and inversely the highest significance, therefore it is entered into the model.

Step 1 Enter the RG variable

Each of the three remaining variables is entered into the model (in addition to the RG variable) and again the statistics are compared. In this case the number of flares is clearly the most significant variable (in actuality the only significant variable) so it enters the model.

Table 2

Remaining Candidate Variables After Step 1

Variable	Chi-Square	Prob>Chi-Square
IR	0.6652	0.4147
CHAFF	0.6652	0.4147
FLARES	2.5717	0.1088

Step2 Enter the FLARES variable

The two remaining variables are entered into the current model, but neither proves to be significant, and both exceed the 0.25 threshhold, so the algorithm stops.

Table 3

Remaining Candidate Variables After Step 2

Variable	Score Chi-Square	Prob>Chi-Square
IR	0.8721	0.3504
CHAFF	0.8721	0.3504

The test statistics and the associated probabilities show the RG variable to be highly significant, and the FLARES variable to be only marginally significant. This yields the following model

$$\pi(x) = \frac{e^{\beta_0+\beta_1 RG+\beta_2 FLARES}}{1+e^{\beta_0+\beta_1 RG+\beta_2 FLARES}} = \frac{e^{-4.2163+0.1823_1 RG+0.0672 FLARES}}{1+e^{-4.2163+0.1823 RG+0.0672 FLARES}}. \quad (32)$$

Given the marginal significance of the FLARES variable and our desire for a parsimonious model, the next step is to compare this model against the model without the FLARES variable, which in this case is the univariate model based on the RG variable. We compare the models on the basis of the log likelihood and on their predictive capabilities.

Table 4

Comparison of -2 Log-Likelihood

for

RG Model vs RG and FLARES Model

Model	Degrees of Freedom	-2 Log L Intercept Only	-2 Log L Intercept and Ind. Variables	Chi-Square Statistic	Prob>Chi-Square
RG	1	277.148	266.669	10.449	0.0012
RG and FLARES	2	277.144	264.145	13.0003	0.0015

Recall that by multiplying the log-likelihood by -2 the resulting statistic is distributed under the chi-square distribution. In table 4 the chi-square statistic in the fifth column is the difference between the third and fourth column. This represents the difference in the log-likelihood due to the inclusion of the independent variables, under the null hypothesis that the fit is adequate. The sixth column is the probability that we will find a statistic of this magnitude or larger if that hypothesis is true, (i.e that it occurred due to random chance and the parameters actually equal zero). Comparing the log-likelihood ratios for the two models we find little difference between the multivariate model with the RG and FLARES varaibles and the univaraite model contianing only the RG variable. However, in both case the low probabilities indicate that we should reject the assumption of an adequate fit. (In the case of both of these models, the prediction performance is abysmal, which reflects the poor fit. Out of 640 trials, 36 of which are successes, both models classify all predictions as failures. Further investigation reveals that the multivariate model is marginally better, but overall neither has any utility for predictive purposes.)

This leads us to examine the diagnostics for the models, in particular those diagnostics which will help us identify observations which are associated with a poor fit. These include DIFDEV, DIFCHISQ, Pearson's residual and the deviance residual, and the hat matrix diagonal. The first two of these, DIFCHISQ and DIFDEV provide us insight into the effect each observation has on the chi-square statistic and the deviance, respectively.

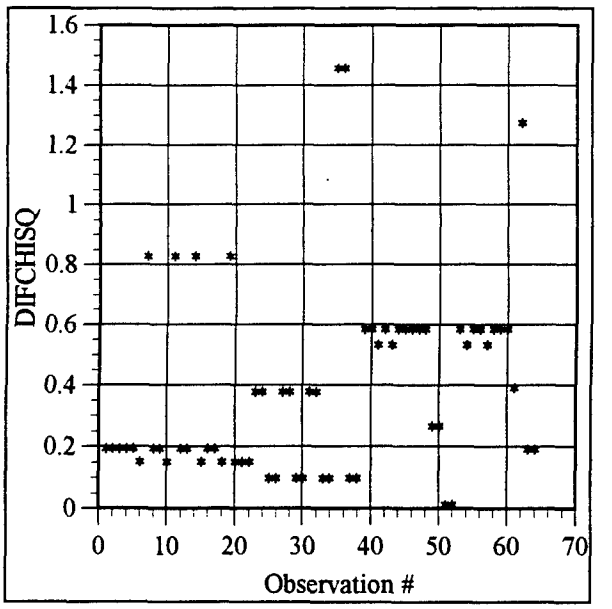


Figure 1 - Plot of DIFCHISQ versus Observation Number.

Examining figure 1, it clear that three of the observations have an inordinate influence on the chi-square statistic. Observations #35 and #36 are grouped together as a point #15 in the experimental design. Comparing the data points reveals that this point has the minimum possible number of RG missiles and has the other variables at or near their respective maximum values. In contrast, observation #62 also appears to be problematic, but it has no flares, and median values for the other three variables. Note that it is grouped with observation #61, but #61 is a failure while #62 is a success.

In contrast the chi-square statistic, the deviance was primarily affected by observations #7, #11, #14, and #19. Each of these is located near the edge of the design space, as opposed to the center, and each is paired with another observation at its respective design point. At each of these points there is one success and one failure, and the four points listed above are all failures.

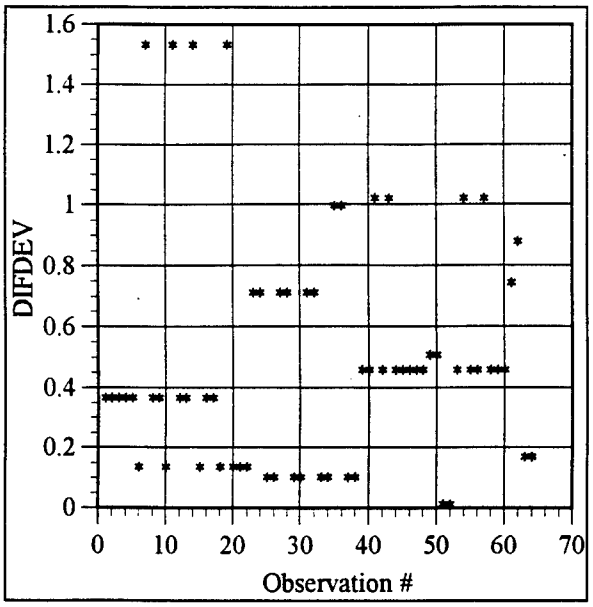


Figure 2 - Plot of DIFDEV versus Observation Number

Overall, in the DIFCHISQ and DIFDEV graphs, there are only a small percentage of point which are extreme, and a small number of moderately high values. It might be worthwhile to examine the extreme points and perform the analysis with them deleted, and compare the results.

Those observations which the hat matrix diagonal identifies as extreme points in the design space, do not coincide with those identified in the previous two figures. The relatively high percentage of extreme points and moderately extreme points indicate a need to evaluate the experimental design from the standpoint of logistic metamodelling (as opposed to least-squares modelling, which is how the experiment which generated this data was originally designed).

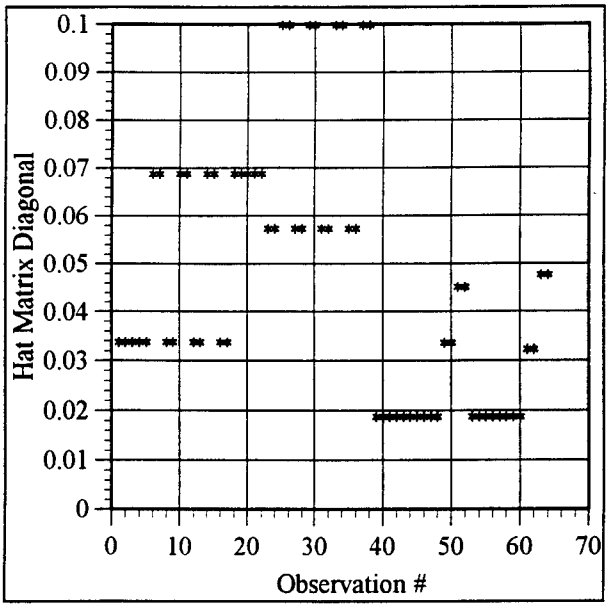


Figure 3 - Plot of Hat Matrix Diagonal versus Observation Number

The Pearson's chi-square residual and devaince residual are useful for identifying observations which are not well accounted for by the model. In Figures 4 and 5, the relatively high proportion of observations with moderate to high values in the thes two diagnostic measure reflect our earlier conclusion that the goodness of fit is inadequate. Again, observations #34, #35, and #62 stand out from the other observations.

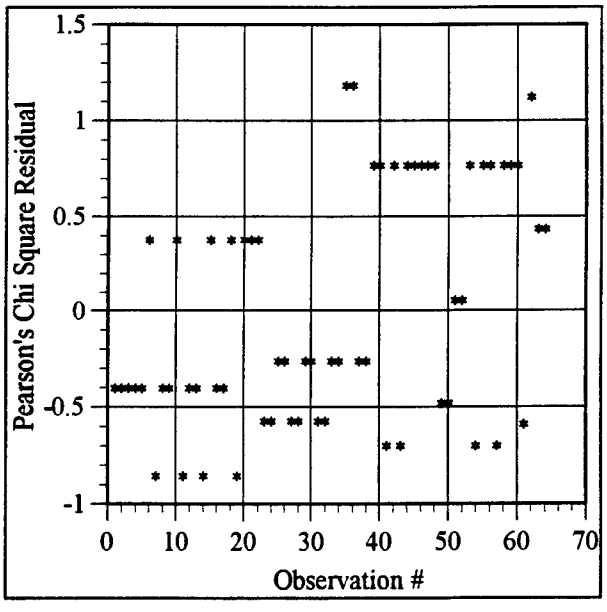


Figure 4 - Plot of Pearson's Chi-Square Residual versus Observation Number

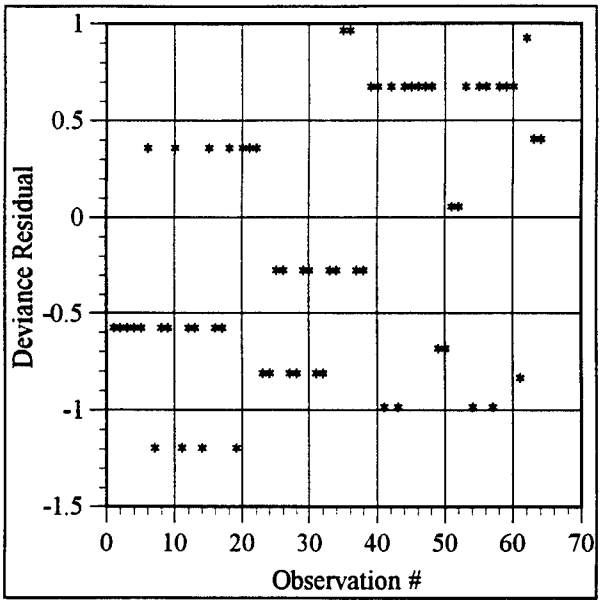


Figure 5 - Plot of Deviance Residual versus Observation Number

4. Conclusions

In the modelling of multiplane aircraft engagements, a number of tools are available for the analysis of the output. One of the more useful methodologies is metamodelling, which to date primarily has focused on least-squares and general linear models. In many cases however the response or performance metric of interest can be characterized as a dichotomous or an ordinal polytomous variable. Whereas linear models are generally not appropriate nin such cases, logistic metamodelling offers an alternative which is well suited for modelling probabilities and proportions in terms of success or failure.

In this report we discussed much of the background in the development of logistic metamodels. This included the assumptions, the underlying parametric model, the basic development methodology, and some of the available tools for evaluating the resulting model. The material in this report is not definitive or comprehensive, but rather provides an introduction to the methods and some of the available references.

To date the majority of work in logistic analysis and metamodelling have focused on univariate models. Recently published and currently ongoing research (doctoral dissertations) are exploring the area of efficient experimental designs for multivariate applied logistic models. One promising avenue of future

research is the application of these results in the area of metamodelling, particularly in applied problems where the high dimensionality of the design space could make it infeasible to conduct an efficient investigation in a timely fashion. A related area of research is to conduct a study of the performance metrics used in the analysis of doctrines and tactics and to assess their ability to convey the information which is available from the models.

Appendix I

This appendix contains the data from the TAC BRAWLER example. The original data came from Tew and Zeimer (1993), in an application of least-squares metamodelling. Each of the 64 observations represents the results of 10 independent trials, and is grouped with respect to its point in the central composite design. The factors in the last four columns reflect the external stores configuration for each of the blue planes in the scenario.

Observation	Group	Avg Kill Ratio	Number of RG Missiles	Number of IR Missiles	Number of Chaff Cartidges	Number of Flares
1	1	1.143	1	1	0	0
2	1	0.529	1	1	0	0
3	1	1.000	1	1	0	0
4	1	0.706	1	1	0	0
5	1	1.214	1	1	0	0
6	2	3.000	9	1	0	0
7	2	1.308	9	1	0	0
8	3	1.000	1	9	0	0
9	3	1.231	1	9	0	0
10	4	1.750	9	9	0	0
11	4	1.214	9	9	0	0
12	5	0.933	1	1	10	0
13	5	0.611	1	1	10	0
14	6	1.308	9	1	10	0
15	6	1.700	9	1	10	0
16	7	0.786	1	9	10	0
17	7	1.500	1	9	10	0
18	8	3.837	9	9	10	0
19	8	0.867	9	9	10	0
20	8	2.333	9	9	10	0
21	8	3.667	9	9	10	0
22	8	5.200	9	9	10	0
23	9	0.882	1	1	0	10
24	9	0.611	1	1	0	10
25	10	3.143	9	1	0	10
26	10	5.000	9	1	0	10
27	11	0.714	1	9	0	10
28	11	1.071	1	9	0	10
29	12	1.667	9	9	0	10
30	12	2.100	9	9	0	10
31	13	0.688	1	1	10	10
32	13	0.750	1	1	10	10
33	14	1.727	9	1	10	10
34	14	2.444	9	1	10	10
35	15	1.727	1	9	10	10
36	15	1.583	1	9	10	10
37	16	1.667	9	9	10	10
38	16	3.429	9	9	10	10

Observation	Group	Avg Kill Ratio	Number of RG Missiles	Number of IR Missiles	Number of Chaff Cartidges	Number of Flares
39	17	1.545	5	5	5	5
40	17	3.000	5	5	5	5
41	17	1.200	5	5	5	5
42	17	2.625	5	5	5	5
43	17	1.500	5	5	5	5
44	17	14.000	5	5	5	5
45	17	2.111	5	5	5	5
46	17	2.000	5	5	5	5
47	17	2.444	5	5	5	5
48	17	7.000	5	5	5	5
49	18	0.923	1	5	5	5
50	18	1.000	1	5	5	5
51	19	2.300	9	5	5	5
52	19	3.000	9	5	5	5
53	20	2.222	5	1	5	5
54	20	1.154	5	1	5	5
55	21	1.727	5	9	5	5
56	21	2.200	5	9	5	5
57	22	0.933	5	5	0	5
58	22	1.600	5	5	0	5
59	23	1.636	5	5	10	5
60	23	2.000	5	5	10	5
61	24	1.231	5	5	5	0
62	24	2.875	5	5	5	0
63	25	1.700	5	5	5	10
64	25	2.875	5	5	5	10

Bibliography

1. Abdelbasit, K. M. and Plackett, R. L. , "Experimental Design for Binary Data", *Journal of the American Statistical Association*, **78**, 90-98,1983
2. Bendel, R. B. and Afifi, A. A., "Comparison of Stopping Rules in Forward Regression", *Journal of the American Statistical Association*, **72**, 46-53, 1977.
3. Box and Draper, 1981. *Applied Linear Regression*, John Wiley and Sons, New York.
4. Feinberg, S. E., 1985. *The Analyis of Cross-Classifed Categorical Data*, MIT Press, Cambridge MA.
5. Hicks, C. R., 1982. *Fundamental Concepts in the Design of Experiments*, Holt, Rinehart, and Winston, Inc., New York, NY
6. Heise, M. A., 1993, "*Optimal Designs for Bivariate Logistic Regression Model*", Unpublished Ph.D. dissertation, Virginia Polytechnic Institute and State University, Department of Statistics, Blacksburg, VA
7. Hosmer, D. and Lemeshow, S., 1989. *Applied Logistic Regression*, John Wiley and Sons, New York, NY.
8. Law, A. and Kelton, D., 1991. *Simulation Modeling and Analysis*, McGraw Hill, New York, NY
9. Minkin, S., " Optimal Designs in Binary Data", *Journal of the American Statistical Association*, **82**, 1098-1103,1987
10. Mickey, J. and Greenland, S. , "A Study of the Impact of the Confounder-Selection Criteria on Effect Estimation", *American Jounal of Epidemiology*, **129**, 125-137, 1989.
11. Myers, R. H., 1991. *Classical and Modern Regression with Applications*, PWS-Kent, Boston, MA.
12. Myers, W. R., 1991. "Optimal Experimental Designs for Fitting the Logistic Regression Model", unpublished Ph.D. dissertation, Virginia Commonwealth University, Department of Biostastistics, Richmond, VA
13. Pregibon, D., "Logistic Regression Diagnostics", *Annals of Statistics*, **9**, 705-724, 1981.
13. *SAS/STAT User's Guide, Volume 2*, Chapter 27 , 1989, SAS Institute, Cary, NC
14. Zeimer, M. and Tew, J, "Metamodelling Techniques and Parametric Statistical Methods for Use With TAC BRAWLER" , Technical Report, Virginia Polytechnic Institute and State University, Department of Industrial and Systems Engineering, 1993

AUTOMATIC DETECTION OF PROMINENCE IN SPONTANEOUS SPEECH

Colin W. Wightman
Assistant Professor
Department of Electrical Engineering

New Mexico Institute of Mining and Technology
Socorro, NM 87801

Final Report for
Summer Research Extension Program
AFOSR Contract No. F4962090C09076

Sponsored by:
Air Force Office of Scientific Research
Bolling Air Force Base, Washington, D.C.
and
New Mexico Tech

December 1993

AUTOMATIC DETECTION OF PROMINENCE IN SPONTANEOUS SPEECH

Colin W. Wightman
Assistant Professor
Department of Electrical Engineering
New Mexico Institute of Mining and Technology

Project Summary

Prominences, which occur when a speaker emphasizes a word so that it "stands out" to the listener, play a number of important roles in aiding the listener in the interpretation of speech. In particular, prominences are crucial for signaling many of the discourse events which conversants use to coordinate a conversation. However, to develop automated algorithms to make use of prominences in automatic speech processing systems, an algorithm which can consistently detect prominences in spontaneous speech must be developed. In previous work, we investigated the ability of untrained human subjects to consistently label prominences in spontaneous speech.

In this project, we sought to develop an algorithm which can consistently label prominences in spontaneous speech. We began by utilizing an algorithm which we had previously developed and used successfully for labeling intonational features in read speech. It quickly became apparent however, that the performance of this algorithm is critically dependent on the availability of accurate segmental transcriptions of the utterances being labeled. The generation of these transcriptions, however, is time-consuming and difficult. Consequently, the bulk of our time in this project was spent developing a tool to address this problem rather than on the detection of prominences. The resulting tool is both powerful and efficient, and has provided the initial idea for a commercial product which will be developed in the near future.

AUTOMATIC DETECTION OF PROMINENCES IN SPONTANEOUS SPEECH

Colin W. Wightman

1 Introduction

We are concerned with the detection of prominence, which occurs when a speaker emphasizes a word or syllable so that it "stands out" to the listener. Prominences are generally used to focus the listeners attention and mark new or important information. Automatic detection of prominences may thus aid the interpretation of an utterance by identifying key words and facilitating semantic disambiguation. At a higher level, prominences play a central role in marking discourse structures which conversants use to identify shifts in topic, corrections, and sub-dialogues. For example, Chen and Withgott [1] have used prominences to automatically select summarizing excerpts from spontaneous speech. It has also been claimed that words which contain a prominence are more carefully articulated than other words [3]. If this is the case, then automatic detection of prominence could help detect these "islands of reliability" and thus improve recognition performance.

In spoken English, prominences provide the listener with information related to the semantic content of an utterance and mark significant discourse structures. This observation, however, has had a relatively small influence on the design of current speech understanding systems: prominence remains essentially unused in current speech-based systems. This has occurred for two reasons: (1) the linguistic community has not produced a unified theoretical framework which makes explicit the role of prominence in conveying syntactic, semantic, and discourse-level information, and (2) automatic methods to reliably detect prominences have not been available to developers. Both of these problems have been exacerbated by a lack of speech corpora with consistently labeled prominences. Without such corpora, neither linguistic research, nor algorithmic

development can proceed efficiently.

We sought to address this by extending the intonational feature labeling algorithm developed by Wightman and Ostendorf for professionally read speech [5] to spontaneous speech by non professional speakers. The Wightman-Ostendorf algorithm works by utilizing a phonetic segmentation of the utterance and extracting an acoustic feature vector for each syllable. These feature vectors are then quantized via a tree quantizer and the resulting codewords used as the observation sequence in a Hidden Markov Model. The HMM is fully connected and has four states, corresponding to various intonational features such as prominence. Viterbi decoding is used to recover the intonational labels from the observed codeword sequence. This architecture is highly extensible: we expected the modifications needed to handle spontaneous speech to be limited to changes in the features which are extracted, and re-estimation of the model parameters.

2 Progress

During preliminary experiments to evaluate the performance of the baseline Wightman-Ostendorf algorithm, it became apparent that the performance of the algorithm was being severely limited by the quality of the data being used to generate the acoustic feature vectors. Specifically, the KING database[1] does not include sufficiently consistent segmental labeling.

While the database does include a phonetic labeling, the phones used are highly variable polyphone units rather than monophones resulting in a very small number of occurrences of a very large number of phones-units. This virtually precluded estimating parameters such as the mean and variance of duration for each phone because the number of occurrences was so low. Moreover, the vowel segments were not marked for stress. These two limitations severely reduced the performance

[1]The speech corpus used in this project, often referred to as the KING database [2], consists of excerpts from telephone conversations in which the subjects were asked to describe various objects and pictures over the phone.

of the Wightman-Ostendorf algorithm in which: (1) the duration of each phone must be normalized by its mean and variance to measure duration lengthening, and (2) previous work has shown that the single most important feature for detecting prominences is the presence or absence of stress on a vowel segment [4].

As a result of these observations, it was determined that better segmental transcriptions would be needed before further work on automatic detection could proceed. Consequently, work began to develop a software tool which would allow researchers to quickly produce accurate phonetic transcriptions at the monophone level with stress-marked vowels.

In the following subsections we briefly review the preliminary experiments with the Wightman-Ostendorf Algorithm, and the key ideas of the software tool developed.

2.1 Preliminary Experiments

Although the Wightman-Ostendorf algorithm was developed using speech produced by professional radio news announcers reading from scripts, we began by investigating its utility in other environments by using a corpus of spontaneous speech. We used the excerpts from the first nine speakers of the "San Diego" subset of the KING corpus, which yielded nine excerpts containing a total of roughly seven minutes of speech. The prominences in these excerpts were then labeled by a panel of eight naive subjects. By declaring a syllable prominent if three of the eight panel members labeled it as prominent, a transcription which exhibited good agreement with a transcription produced by an "expert" labeler was obtained. Overall, the panel and the expert were in agreement 88.2% of the time: 77% of the prominences were correctly detected by the panel, and the false detection rate was 8.1%.

To evaluate the Wightman-Ostendorf algorithm on this corpus, the hand-generated phonetic

	unmarked	prominent
unmarked	679	118
prominent	144	99

Table 1: A confusion matrix showing the relationship between labels produced by the panel of human subjects and the labels generated automatically. The rows correspond to the hand-labeled syllables, while the columns correspond to the automatically produced labels.

transcription provided as part of the corpus was used. This led to the shortcoming described above. Except for these, all other features were extracted in precisely the same manner as for read speech. The speech corpus was divided into three parts and the labeling parameters were estimated using two thirds, while the remaining third was used as test data. This was repeated three times, using a different third of the corpus for testing each time.

The results of this first study are shown in Table 1. When the Wightman-Ostendorf algorithm was applied to read speech, it achieved a correct detection rate of 78% with a false alarm rate of 13%. As can be seen from Table 1., the false alarm rate has remained virtually unchanged at 15%, but the detection rate has dropped to only 41%. Some of the missed detections are not serious in terms of the applications discussed above: if we are interested in detecting *words* which are prominent, labeling the wrong syllable within the correct word should not pose a problem. Nevertheless, if we evaluate the algorithm in terms of correct word-level labeling, the correct detection rate increases only to 49%, still leaving half of the prominences undetected.

We can draw some preliminary conclusions from these results. The first is that, while the prominence detection rate is 53% lower in spontaneous speech than in read speech, the false alarm rate is virtually unchanged. This suggests that, while the differences between read and spontaneous speech make detection of prominences more difficult, they do so only by obscuring or modifying the prominences, not by inserting spurious ones. This is an encouraging result because it suggests

that the unmodified Wightman-Ostendorf algorithm provides sufficient rejection of non-prominent syllables, allowing us to focus on improving its detection capabilities in spontaneous speech.

2.2 An Alignment Tool

The software tool developed to help generate accurate phonetic transcriptions of spontaneous speech has several features:

- It is interactive. The user sees the utterance waveform in a window and uses markers to delimit a section to be aligned, enters the transcription of that section, and commands the tool to generate the alignment.

- It does not require a phonetician. Because the transcription is entered at the word level, a detailed knowledge of phonetics is not needed: the software retrieves all possible pronunciations of each word from the dictionary and automatically determines the best phone-level transcription.

- It is based on existing toolkits for signal processing and HMM modeling. By using the WAVES, ESPS, and HTK toolkits from Entropic Research Laboratory, a modular design was achieved with a minimum of new code.

- It is reasonably accurate. The phone-level HMM models were trained on the TIMIT corpus, and thus are reasonably robust to variations in speakers.

- It marks stressed vowels. Bill Fisher from NIST provided a set of TIMIT label files which had been augmented with stress markings. These were used to develop separate models for stressed and unstressed productions of each vowel. During the alignment process, both the stressed and unstressed vowel models are used for each syllable which is marked in the

dictionary as possibly receiving stress. The recognition search then determines if the stressed or unstressed model accounts for the data better.

The alignment tool has generated considerable interest and is being transfered to Entropic Research Laboratory for further development and possible release as a commercial product.

3 Professional Communications

The following set of relevant professional communications would not have been possible without the funding provided by this project. We append the paper and abstract that have resulted.

- C. W. Wightman. "Annotation of Prominence in Spontaneous Speech". In *Proceedings 1993 Mohawk Valley IEEE Dual Use Technologies and Applications Conference*, Utica, NY.
- C. W. Wightman. "Perception of Multiple levels of Prominence in Spontaneous Speech" (abstract only). Poster presented at the fall meeting of Acoustical Society of America, Denver, 1993.

References

[1] F. Chen and M. Withgott. "The use of emphasis to automatically summarize a spoken discourse". In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing"*, pages I–229, 1992.

[2] A. Higgens, E. Wrench, L. Bahler, J. Porter, D. Schmoldt, and M. Lipps. "Speaker identification and recognition", Final Report, Contract 88-F744200-000. Technical report, ITT Aerospace/Communications, San Diego, CA, 1991.

[3] W. Lea. "Prosodic aids to speech recognition". In W. Lea, editor, *Trends in Speech Recognition*, pages 166–205. Prentice-Hall, Inc., 1980.

[4] C. Wightman and M. Ostendorf. "Automatic labeling of prosodic patterns". *IEEE Trans. on Speech and Audio Processing*, 1994. To appear.

[5] C. W. Wightman and M. Ostendorf. "Automatic recognition of intonational features". In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, San Francisco, 1992.

Annotation of Prominence in Spontaneous Speech

Colin W. Wightman*
Department of Electrical Engineering
New Mexico Institute of Mining and Technology
Socorro, NM 87801

March 1, 1993

Abstract

Prominences, which occur when a speaker emphasizes a word so that it "stands out" to the listener, play a number of important roles in aiding the listener in the interpretation of speech. In particular, prominences are crucial for signaling many of the discourse events which conversants use to coordinate a conversation. In this paper, we discuss some of the potential applications of automatic prominence detection methods. Preliminary results from an ongoing effort to extend algorithms developed for use on read speech are presented. Although the correct detection rate is reduced by more than 50% when the read speech algorithm is simply applied to spontaneous speech, the false detection rate remains virtually unchanged at 15%. This suggests that prominences in spontaneous speech are less dramatically marked than in read speech, but that the differences do not have the effect of inserting spurious prominences.

1 Introduction

In spoken English, prominences provide the listener with information related to the semantic content of an utterance and mark significant discourse structures. This observation, however, has had a relatively small influence on the design of current speech understanding systems: prominence remains essentially unused in current speech-based systems. This has occurred for two reasons: (1) the linguistic community has not produced a unified theoretical framework which makes explicit the role of prominence in conveying syntactic, semantic, and discourse-level information, and (2) automatic methods to reliably detect prominences have not been available to developers.

We are concerned with the detection of prominence, which occurs when a speaker emphasizes a word or syllable so that it "stands out" to the listener. Prominences are generally used to focus the listeners attention and mark new or important information. Automatic detection of prominences may thus aid the interpretation of an utterance by identifying key words and facilitating semantic disambiguation. At a higher level, prominences play a central role in marking discourse structures which conversants use to identify shifts in topic, corrections, and sub-dialogues. For example, Chen and Withgott [3] have used prominences to automatically select summarizing excerpts from spontaneous speech. It has also been claimed that words which contain a prominence are more carefully articulated than other words [5]. If this is the case, then automatic detection of prominence could help detect these "islands of reliability" and thus improve recognition performance.

In recent work, Wightman and Ostendorf [10, 8] have reported on an automatic algorithm for labeling prominences. While their algorithm has achieved good accuracy when used to label speech produced by professional radio announcers reading scripts, it has not been evaluated on spontaneous speech. Spontaneous speech, produced when the speaker is formulating the communicative plan and producing speech simultaneously, contains a number of phenomena which do not occur in read speech. In particular, some of the acoustic features used to convey information above the word-level (pausing, for example) appear to be used differently in the two types of speech. In this paper, we examine the motivation for developing prominence detection methods for spontaneous speech, and present results from an experiment in which the Wightman-Ostendorf algorithm was applied to spontaneous speech.

2 Potential Applications

In conversation, human listeners easily follow changes in speaker, shifts in topic, interruptions, corrections, and asides. Current speech understanding systems, however, do not have such capabilities. In large measure, this is due to the fact that they do not have access to the cues which signal significant discourse events: the speaker's prosody. Most current systems avoid this problem by restricting the user to a small task and constraining the application domain so that any uncertainty about the topic of an utterance could be trivially resolved. In the Air Travel Information Service (ATIS) demonstration systems (c.f. [7]), for example, the user is assumed to be talking only about a single, direct flight, and is limited to a simple ques-

*This work was supported in part by the Air Force Office of Scientific Research, Bolling AFB, Washington, D.C., under grant number F49620-90-C-09076 and the Summer Faculty Research program.

tion/response interaction. As systems for more complex tasks and with more "natural" speech interfaces are developed, however, the need for discourse analysis becomes critical for providing robust interfaces which the user can engage in a more familiar way. For an automated system to be able to carry out effective discourse analyses, however, detection of the prominences which signal many of the significant discourse events will be needed.

A great deal of conversational speech is devoted to corrections, clarifications, and confirmations, as the conversants work together to ensure that the ideas are correctly transmitted. Prominence plays a central role in this process, marking key words which signpost the dialogue. For example, a speaker might say "I'm going to eat with *Boston* ... *in* Boston." The first occurrence of *Boston* is emphasized because it is new information. The speaker then realizes their error and repeats the clause emphasizing *in* to mark the repeat as a correction. Notice that, had the speaker corrected their error faster, the prominence would be the only way of detecting this correction: "I'm going to eat with ... *in* Boston".

As designers of speech understanding systems, we need to determine if any of the roles prominence plays in normal, human to human, communications are of relevance to communications between humans and machines. In particular, we need to identify ways in which automated detection of prominence could improve the ability of speech processing systems to fulfill their missions. From this perspective, one can identify three general roles for automatic detection of prominence: (1) coding prominences for speech compression/conversion, (2) detecting prominences for discourse analysis, and (3) labeling prominences in large research corpora.

The most straightforward application of a prominence detection algorithm would be in speech coding. For example, one approach to extremely narrowband coding (*e.g.* 50 baud), is to use a speech recognizer on the transmitting end, send codes to identify the words recognized, and then resynthesize the words at the receiving end. While this method yields perfect intelligibility, it destroys all of the prosodic information in the speech. Automatic detection of prominence would permit each word to be tagged with a one-bit flag indicating whether that word should be resynthesized as prominent or not. The simplicity of this application comes from the fact that the system need only be able to detect and synthesize prominences: all of the discourse processing cued by the prominences is done by the users of the system.

This technique could substantially enhance the utility of the narrowband channel by allowing the users to coordinate their discourse activities in a more normal manner. Its application is not limited to narrowband communications, however: it could also be used whenever speech information needs to be stored compactly, such as in talking dolls, or electronic games.

A similar application would be to integrate prominence into a voice translation system which translates an utterance spoken in one language to another language. Again, this is done by using a speech recognizer, some translation process, and synthesizing the output. And, as with narrowband coding, adding prominence detection and synthesis would not require understanding all of the ways prominence is used in coordinating a dialog: the users provide that function. What would be required is some modification to the translation process to make sure that the prominence gets resynthesized in the right place. If the speaker stresses the verb of a sentence, for example, the system's output should stress the verb too, even though it may appear in a completely different place in the sentence. Such a system could be used for a variety of purposes including a translation aid for deaf persons, in which the input speech was transcribed, and prominent words displayed in bold type.

A much more ambitious application would be a system which actually analyzes the speaker's discourse using the prominences as a cue. Applications for this sort of system abound in both the civilian and military sectors. From automated information retrieval systems, to full language translations, to conversational workstations, to systems which can monitor and summarize complex communications, such systems will require not only an automatic detection algorithm, but also a detailed understanding of how prominence can be used to identify the desired information in a conversation. Such an understanding can only come, however, from further research into the roles of prominence, which brings us to the third general role for automated prominence detection: labeling research corpora.

Researching the roles of prominence will require the use of large speech corpora (many hours) in which prominences have been labeled. Labeling prominences in such a large corpus, however, is a daunting task. While hand-labeling can be quite accurate, it is also extremely time-consuming: in the test described in [9], eight person-hours were required to transcribe less than ten minutes of speech. Clearly, labeling of the large corpora required will necessitate the application of an automated labeling algorithm. Even if the labels produced by the algorithm need to be hand corrected in a second pass, this is still likely to be far faster than labeling by hand. Indeed, this is the approach used in the University of Pennsylvania's TREEBANK project [6] which efficiently annotates syntactic bracketing in very large corpora.

3 The Read Speech Algorithm

The Wightman-Ostendorf algorithm [8] was developed for labeling a variety of prosodic features including prominences. In essence, a speech recognizer is used to provide word and phoneme alignments for an utterance, which are then used to produce feature vectors from the speech signal. These vectors, which may include categorical as well as numerical features, are then mapped to a set of code-

words via a decision tree. Although the decision tree is designed using standard algorithms [2], it is not used to classify the labels directly, but rather to provide observations for a discrete hidden Markov model. Each state of the hidden Markov model corresponds to a class to be recognized, and therefore standard Viterbi decoding is used to recover the labels which correspond to the underlying state sequence.

Although the original algorithm labeled some intonational features in addition to prominences, we have a relatively small amount of training data and consequently chose to label only two categories: P and S (corresponding to syllables which are and are not perceived as prominent). Since we have only two possible labels, the hidden Markov model will have only two states. By combining categories in this way, we obtain more training examples for each category, which may result in a more robust model. According to linguistic theory [1], pitch accents (one form of prominence) are assigned to syllables and so we will extract one feature vector for each syllable. The sequence of feature vectors will then be quantized and decoded to a sequence of syllable labels.

In order to extract one feature vector per syllable, it is necessary to determine the syllable boundaries. This is a potentially difficult task since syllable structure is so varied and often ambiguous. For this pilot study, however, the number of syllables is relatively limited, and we chose to simply mark them by hand. For larger tasks, syllable boundaries would be determined automatically according to the dictionary used by the speech recognizer during the labeling of the phonetic segmentation. Given the syllable and phonetic boundaries, a set of twelve features are extracted for each syllable (see [10] for a detailed description of the feature set). Although pitch accents are primarily intonational features, they coincide with other acoustic cues to the prominence phenomena that they mark. Therefore, duration lengthening, pauses and energy changes are also used as features.

4 Experimental Results

Although the Wightman-Ostendorf algorithm was developed using speech produced by professional radio news announcers reading from scripts, we have begun investigating its utility in other environments by using a corpus of spontaneous speech. The corpus, often referred to as the King database [4], consists of excerpts from telephone conversations in which the subjects were asked to describe various objects and pictures over the telephone. The subjects were recorded both locally, and over the telephone line which makes it possible to evaluate a system on the same speech under both clean and noisy conditions. Roughly fifty speakers took part in the test which was done in ten sessions producing five hundred excerpts of between forty and sixty seconds each.

For our evaluation of prominence labeling, we utilized a subset of this corpus drawn from the first session. We used the excerpts from the first nine speakers, which yielded nine excerpts containing a total of roughly seven minutes of speech. The prominences in these excerpts were then labeled by a panel of eight naive subjects. By declaring a syllable prominent if three of the eight panel members labeled it as prominent, a transcription which exhibited good agreement with a transcription produced by an "expert" labeler was obtained. Overall, the panel and the expert were in agreement 88.2% of the time: 77% of the prominences were correctly detected by the panel, and the false detection rate was 8.1%.

	unmarked	prominent
unmarked	679	118
prominent	144	99

Table 1: A confusion matrix showing the relationship between labels produced by the panel of human subjects and the labels generated automatically. The rows correspond to the hand-labeled syllables, while the columns correspond to the automatically produced labels.

To evaluate the Wightman-Ostendorf algorithm on this corpus, a hand-generated phonetic transcription was used. This was done because of difficulties in generating the transcription using available recognizers. As a result, the phonetic labeling used did not distinguish between stressed and unstressed instances of a given vowel, a distinction which is used as a feature in the Wightman-Ostendorf algorithm. Except for this, all other features were extracted in precisely the same manner as for read speech. The speech corpus was divided into three parts and the labeling parameters were estimated using two thirds, while the remaining third was used as test data. This was repeated three times, using a different third of the corpus for testing each time.

The results of this first study are shown in Table 1. When the Wightman-Ostendorf algorithm was applied to read speech, it achieved a correct detection rate of 78% with a false alarm rate of 13%. As can be seen from Table 1., the false alarm rate has remained virtually unchanged at 15%, but the detection rate has dropped to only 41%. Some of the missed detections are not serious in terms of the applications discussed above: if we are interested in detecting *words* which are prominent, labeling the wrong syllable within the correct word should not pose a problem. Nevertheless, if we evaluate the algorithm in terms of correct word-level labeling, the correct detection rate increases only to 49%, still leaving half of the prominences undetected.

5 Conclusions

We can draw some preliminary conclusions from these results. The first is that, while the prominence detection rate is 53% lower in spontaneous speech than in read speech, the false alarm rate is virtually unchanged. This suggests that, while the differences between read and spontaneous speech make detection of prominences more difficult, they do so only by obscuring or modifying the prominences, not by inserting spurious ones. This is an encouraging result because it suggests that the unmodified Wightman-Ostendorf algorithm provides sufficient rejection of non-prominent syllables, allowing us to focus on improving its detection capabilities in spontaneous speech.

One source of improved performance will be the use of automatically generated phonetic transcriptions which include marking of stressed vowels. In read speech corpora, the distinction between stressed and unstressed vowels was found to be the most important single feature for labeling prominences. We therefore expect that marking stressed vowels in the King database will provide substantially improved performance. In addition, the hand-generated phonetic transcriptions used in this study used biphone and triphone labels inconsistently, resulting in some anomalous feature vectors. Although the impact of these anomalies on the labeling performance is unclear, the use of automatically generated transcriptions should address this issue as well. Finally, the data used in this initial investigation represented only a small subset of the King database. Hand-labeling of a larger subset will provide a larger set of training data resulting in more robust estimates of the model parameters.

Despite the limitations of this preliminary work, we have demonstrated the feasibility of labeling prominences automatically in spontaneous speech. By utilizing a modified feature set, the Wightman-Ostendorf algorithm may be able to achieve spontaneous speech performance equivalent to its read speech performance. Such an achievement would enable the development of applications such as those discussed above, opening substantial markets in both the military and civilian sectors.

6 Acknowledgments

The author would like to thank Jim Cupples of the Rome Laboratories Speech Processing Laboratory for his assistance and support of this work. Thanks also to Tim Daniel for providing the King Speech Database.

References

[1] M. Beckman. *Stress and Nonstress Accent.* Foris Publications, Dordrecht, Holland, 1986. Chapter 3.

[2] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth and Brooks/Cole Advanced Books and Software, Monterey, CA., 1984.

[3] F. Chen and M. Withgott. "The use of emphasis to automatically summarize a spoken discourse". In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing"*, pages I–229, 1992.

[4] A. Higgens, E. Wrench, L. Bahler, J. Porter, D. Schmoldt, and M. Lipps. "Speaker identification and recognition", Final Report, Contract 88-F744200-000. Technical report, ITT Aerospace/Communications, San Diego, CA, 1991.

[5] W. Lea. "Prosodic aids to speech recognition". In W. Lea, editor, *Trends in Speech Recognition*, pages 166–205. Prentice-Hall, Inc., 1980.

[6] M. Marcus and B. Santorini. "Building a very large natural language corpora: The penn treebank". Submitted manuscript.

[7] M. Wang and J. Hirschberg. "Predicting intonational boundaries automatically from text: the ATIS domain". In *Proc. Fourth DARPA Speech and Natural Language Workshop*, Asilomar, CA, 1991.

[8] C. Wightman and M. Ostendorf. "Automatic labeling of prosodic patterns". Forthcoming.

[9] C. W. Wightman. "Prominence in spontaneous speech: Annotation and applications". In *AFOSR Summer Research Program, Final Report*, pages 25–1 – 25–17. Air Force Office of Scientific Research, Bolling AFB, Washington, D.C., 1992.

[10] C. W. Wightman and M. Ostendorf. "Automatic recognition of intonational features". In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, San Francisco, 1992.

Perception of multiple levels of prominence in spontaneous speech

Colin W. Wightman (Dept. of Electrical Engr., New Mexico Inst. of Mining and Technology, Socorro, NM 87801)

In both read and spontaneous speech, phrasal prominences play an important role in conveying the speaker's intent. Prominences serve both to mark important discourse-related events in the conversation and help to resolve ambiguities at several levels. Many speech researchers report the intuition that there are several levels of prominence, that is, that some prominences are bigger than others. Nonetheless, attempts to train human labelers to mark multiple levels of prominence have not been successful: while there was agreement on the location of prominences, there was little agreement between labelers on the level to be assigned to each. We have taken an alternative approach, using a panel of naive listeners to mark prominences in a corpus of spontaneous speech. Instead of marking multiple levels of prominence, a simple binary labeling was used by each labeler and the level of each prominence determined by the number of labelers marking it. In this paper, we present the results of this preliminary study, and investigate the relationships between our estimated levels of prominence and the acoustic correlates of vowel lengthening, syllable lengthening, and pitch level and excursion.

Technical Area: Speech Communication
PACS Subject Classification Numbers: 43.71.E, 43.70.F or session on prosody
Telephone number: (505) 835-5708
Telefax number: (505) 835-5707
Method of Presentation: Prefer lecture but willing to give as poster