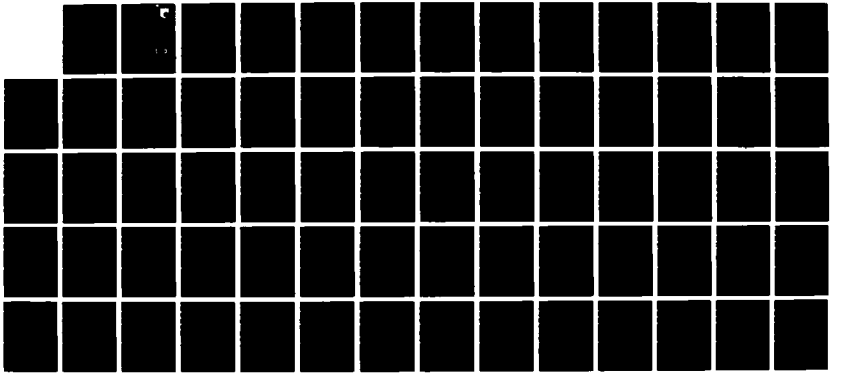
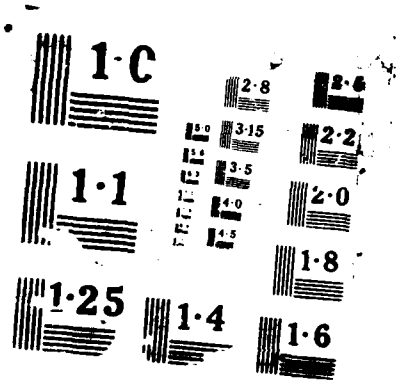
AD-A193 289 A COCKPIT NATURAL LANGUAGE STUDY: VOCABULARY AND GRAMMAR ANALYSES(U) MIDWEST SYSTEMS RESEARCH INC DAYTON OH M P MUNGER ET AL. FEB 88 AFWAL-TR-87-3108 1/1

UNCLASSIFIED F33615-85-C-3623 F/G 5/7 NL




1·C
2·8
3·15
3·5
4·0
4·5
2·2
2·0
1·8
1·1
1·25
1·4
1·6

(2)

AFWAL-TR-87-3108

A COCKPIT NATURAL LANGUAGE STUDY: VOCABULARY AND GRAMMAR ANALYSES

Michael P. Munger, PhD
Ronald L. Small, Captain, USAF
David T. Williamson

MidWest Systems Research, Inc.
1525 Edna Street Dayton, OH 45435

February 1988

Final Report for Period Oct 86 - Sep 87

FLIGHT DYNAMICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6553

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): N/A	5. MONITORING ORGANIZATION REPORT NUMBER(S): AFWAL-TR-87-3108
6a. NAME OF PERFORMING ORGANIZATION: Midwest Systems Research, Inc. / 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION: Crew Systems Development Branch
6c. ADDRESS (City, State, and ZIP Code): 1521 Edna Street, Dayton, OH 45431	7b. ADDRESS (City, State, and ZIP Code): AFWAL/FIGR, Wright-Patterson AFB, OH 45433-6553
8a. NAME OF FUNDING/SPONSORING ORGANIZATION / 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: F-33615-85-C-3623
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
62201F	2403	04	62

11. TITLE (Include Security Classification)

A Cockpit Natural Language Study: Vocabulary and Grammar Analyses

12. PERSONAL AUTHOR(S)

Michael P. Munger, PhD; Ronald L. Small, Capt, USAF; and David T. Williamson

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 86 Oct TO 87 Sep	February 1988	69

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP
05	07	
05	08	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

Avionics, Pilot's Associate, Grammar, Speech Technology, Natural Language, Lexicon, Pilot-Vehicle Interface, Semantics, Parsers

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This report describes recent work which was conducted to facilitate research and development efforts in the use of voice interaction in fighter cockpits. Definitions of an operational vocabulary and grammar are starting points for studying the use of voice interaction in fighter cockpits. This report provides an initial definition of a vocabulary, based on a Pilot's Associate equipped fighter. It also provides software tools for vocabulary, and grammar analysis. Researchers in this area need such tools to adapt quickly to changing technologies as they are applied to cockpits, with consequent changes to vocabulary and grammars. These software tools (1) assist in processing a vocabulary into a lexicon for use by parsers, (2) assist in developing grammars for parsers, and (3) assist in determing semantics from parsers steps.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS	Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL
Capt Ronald L. Small	(513) 255-6644	AFWAL/FIGR

DD Form 1473, JUN 86 — Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

TABLE OF CONTENTS

SECTION	TITLE	PAGE
I	Introduction	1
II	Purpose	1
III	Vocabulary	2
IV	Grammar	2
V	Conclusions	3
Bibliography		5

APPENDICES

APPENDIX	TITLE	PAGE
APPENDIX A	Starting Vocabulary for Pilot's Associate	A-1
APPENDIX B	ANALYZE: Grammar Analysis	B-1
APPENDIX C	VOCAB: Lexicon Preparation	C-1
APPENDIX D	PARSE: Command Parsing	D-1
APPENDIX E	SEMANT: Semantics Extraction	E-1

LIST OF TABLES

TABLE	TITLE	PAGE
Table A-1	Vocabulary Listing	A-3
Table B-1	A Test Grammar	B-5
Table B-2	Screen Messages from ANALYZE	B-6
Table B-3	Input Grammar	B-7
Table B-4	Sorted BNF Vocabulary	B-8
Table B-5	Produced Head Symbols	B-9
Table B-6	C1 Matrix for Stacking Decision	B-10
Table B-7	Context Checks for Equal and Embedded Right Parts	B-11
Table B-8	C2 Production Choice Function	B-13
Table C-1	A Sample Lexicon, Source	C-3
Table C-2	A Sample Lexicon, Converted	C-4
Table D-1	Word Expansion Tables	D-3
Table D-2	Printed Output From PARSE, Terse	D-6
Table D-3	Printed Output From PARSE, Verbose	D-10
Table E-1	Skeleton Procedure for Semantics Extraction	E-2
Table E-2	Printed Output From SEMANT	E-4

I Introduction

Recent studies have been conducted on the premise that speech can have a role in aircraft cockpits, both as an input and as an output means of communication between aircraft and pilot. North and Lea (1982) studied the potential in a large bomber (B-52) while Cotton et al (1983) focused on a modern high-technology fighter (F-16). Moore (1984) suggested some methodologies for designing and studying pilot-vehicle interfaces which employ voice as a means of communication in addition to the usual switches and levers.

Mahendra and Merkel (1986) examined various issues to be encountered in conducting research and development. The result was a series of programmatic recommendations. In general it was suggested that research be conducted in four phases, each of which was to be integrated with on-going Pilot's Associate efforts. These phases were (1) definition of scenarios to be used; (2) initial and limited development and integration of vocabulary, syntax, and grammar; (3) expansion of vocabulary, syntax, and grammar; and (4) full-blown integration with Pilot's Associate. This last phase was to include an extensive integration with pilot intent modelling.

A study by Lizza et al (1987) concentrated on the use of speech as a channel to communicate pilot intentions to the aircraft. A major purpose was to collect words which could be a starting point for defining a vocabulary to be used in subsequent studies.

The general concensus has been that voice could significantly contribute to reduced visual and manual workload and thereby enhance pilot effectiveness and survivability. However, because using voice in a cockpit is such a new area for research and development, there are no well defined results to inform and guide human factors experiments which focus on particular issues, such as reducing error rates.

II Purpose

This report describes recent work which was conducted to facilitate research and development efforts in the use of voice in fighter cockpits. Two inescapable building blocks in such efforts are vocabulary and grammar. It is impossible to construct a voice recognition and understanding system without these two entities being defined and implemented. And it is only with a working system that even minimal investigations can be undertaken.

It is important to note that particular vocabularies and grammars will be strongly influenced by the environment in which they are used. This environment is comprised of a specific cockpit configuration in combination with specific tasks to be performed. If engineering changes are made to a display, for example, it might result in words being added to or deleted from the vocabulary. It might also result in changes in the way the pilot expresses his intentions vis-a vis that display, that is, changes in the grammar.

III Vocabulary

Lizza et al (1987) conducted 54 pilots through a futuristic mission scenario which required them to give verbal commands to their aircraft. The sessions were tape recorded. Subsequently, all the commands, along with identifying information, were transcribed to machine readable form. Selected transcripts are presented in their entirety in Small (1988). Removal of duplicate commands resulted in a file of approximately 2000 commands. This provided a starting point for defining a vocabulary, which, in turn, provided a basis for defining grammars.

As few constraints as possible were placed on the manner in which pilots expressed their verbal commands. Consequently, some commands were verbose, others terse. Some were awkwardly worded, some eloquent. An effort was made to penetrate this variability and find some commonality. An intelligent system, such as Pilot's Associate, was assumed to be helping the pilot with workload, decision aids, and mission information.

The results of this effort are presented in Appendix A. Words were added to cover situations not included in the Lizza et al (1987) study, e.g., pre-takeoff, takeoff, descent, approach, landing, and various emergencies. The vocabulary is presented in a mission phase sequence (rather than alphabetically, by word length, or some other sequence), because it is the way pilots think about using such a sub-system; and it corresponds to the mission flow as represented in the scenario. In addition, only parts of the vocabulary will need to be active during certain phases of the mission. These phases are suggested by the divisions and sub-divisions of Appendix A. This separation into phases could make speech recognition and understanding much easier by restricting the number of alternatives the system must consider. However, to the extent possible, miscellaneous commands need to be active all the time.

IV Grammar

The use of natural language in cockpits requires both speech recognition and speech understanding. Speech recognition can be accomplished by commercially available boards. Output from these boards can be, among other things, an ASCII character string representing the word recognized. Low level software can collect these word strings into a pilot command which is then made available for processing by speech understanding software.

Speech understanding software starts with a string of one or more words and ends with changing the state of the (simulated) aircraft. The programs described here are a collection of tools for developing this software. The "understanding" process is sketched as follows.

The software has available to it a command to be understood and a pre-defined vocabulary. The vocabulary is usually in the form of a lexicon, where each word has stored with it various kinds of information, such as parts of speech, which are useful for analysis. As each word in the command is located in the lexicon, the associated information is made available for parsing.

Parsing is a technique whereby rules of grammar are used to guide the analysis of a sentence (command). These grammar rules are typically expressed in Backus-Naur-Form (BNF) (McKeeman et al, 1970) and incorporate the information contained in the lexicon.

In general, a parser works as follows. Words in the input stream (e.g., a pilot command) are replaced by symbols retrieved from the lexicon. The top of the input stream is removed and placed in a location called the token. If no decisions can be reached based on the top one or two symbols of the parse stack and the token, the token is pushed onto the parse stack and replaced with the next symbol from the input stream. Conceptually, the process looks like the following:

parse stack	token	input stream
... < symbol > < symbol >	< symbol >	< symbol > < symbol > ...

Taken together, the top one or two symbols of the parse stack and the token might match a grammar rule (production, reduction) as described in Appendix B. Such a match is an occasion for the parser to take actions associated with extracting the semantics of the command.

The programs described in this document provide three capabilities: (1) analysis of grammars expressed in BNF, (2) testing grammars by parsing representative commands, and (3) extracting semantics from parsed commands.

All programs are written in Microsoft Pascal (1986). Descriptions of these programs and instructions on how to use them are provided in Appendices B through E. Copies of these programs may be obtained by sending a 5 1/4" MS-DOS compatible diskette to the address given on the cover of this report.

V Conclusions

Definitions of an operational vocabulary and grammar are starting points for studying the use of voice in fighter cockpits. This report has provided an initial definition of a vocabulary. It has also provided software tools for vocabulary and grammar analysis. Researchers in this area need such tools in order to adapt quickly to changing technologies as they are applied to cockpits, with consequent changes to vocabulary and grammars. These software tools (1) assist in processing a vocabulary into a lexicon for

use by parsers, (2) assist in developing grammars for parsers, and (3) assist in determining semantics from parse steps.

Vocabularies, grammars, and even ways of thinking about how to accomplish mission tasks will be greatly influenced by the display devices, information organization, and functional capabilities embodied in future cockpits. In addition to having hardware and software available to conduct voice studies, a researcher must exert considerable effort to estimate what future cockpits will be like. Only with such an estimate or model well in hand can robust voice studies be accomplished.

BIBLIOGRAPHY

Cotton, J. C., McCauley, M. E., North, R. A., & Strieb, M. I. (1983). Development of Speech Input/Output Interfaces for Tactical Aircraft., AFWAL-TR-83-3073, Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Dayton, OH.

Fighter Aircraft Grammar, (1985). Texas Instruments Incorporated, Dallas, TX.

Lizza, G. D., Munger, M. P., Small, R. L., Feitshans, G. L, & Detro, S. D. (1987). A Cockpit Natural Language Study: Data Collection and Initial Data Analysis., AFWAL-TR-87-3003, Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Dayton, OH.

Mahendra, K. & Merkel, P. (1986). Voice Recognition Technology Requirements for Pilot's Associate., BDM/MCL-86-0337-TR, The BDM Corporation, McClean, VA.

McKeeman, W. M., Horning, J. J., & Wortman, D. (1970). A Compiler Generator. Prentice-Hall, Englewood Cliffs, NJ.

MicroSoft Pascal Compiler, (1985). Microsoft Corporation, Bellevue, WA.

Moore, C. A., Moore, R. D., & Ruth, J. C. (1984). Applications of Voice Interactive Systems - Military Flight Test and the Future., American Institute of Aeronautics and Astronautics.

North, R. & Lea, W. A. (1982). Application of Advanced Speech Technology in Manned Penetration Bombers, AFWAL-82-3004, Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Dayton, OH.

Operational Brevity Words, (1985). Tactical Air Command Regulation 55-79, Attachment 2, Langley AFB, VA.

Small, R. L., Flory, D. E., Munger, M. P., Williamson, D. T. & Hollis, B. T. (1988). A Cockpit Natural Language Study: Selected Transcripts, AFWAL-TR-88-XXXX, Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Dayton, OH (in preparation).

APPENDIX A

Starting Vocabulary for Pilot's Associate

Rationale

In the first Cockpit Natural Language (CNL) technical report (Lizza, 1987), the vocabulary data were presented as recorded, in the sequence dictated by the CNL scenario. The vocabulary presented here is "cleaned up" to remove verbose or awkward commands, and commands are added to encompass mission phases or situations not covered in the CNL scenario. Most added commands are based upon comments from the CNL study's subject pilots. Other sources cited are: Texas Instruments' Fighter Aircraft Vocabulary (TI) and Tactical Air Command Regulation 55-79 attachment 2 "Operational Brevity Words " (TAC).

Format

The format for the vocabulary list in Table A-1 is as follows:

Division (e.g., Mission Phase, Formation, Tactics, Miscellaneous)
Sub-Division (Low-level, Air-to-air, Sub-systems)

Commands	Meaning	Source	Response
synonymous commands listed together space left between non-synonymous commands	pilot's intentions	CNL or TI or TAC	terse, meaningful reply; pilot silenceable (non-aural replies in parentheses, e.g., "display change")

Where "Meaning" is obvious, it is left blank.

Assumptions

The assumptions for this vocabulary are that it is intended for a post-1995, single-seat fighter of high performance, with stealth capabilities, full sensor suite for 4-pi

steradian coverage, and equipped with a Pilot's Associate (PA) for intelligent automation. The speech understanding system assumes: 99 + % recognition rate in all flight conditions, continuous speech, real-time response, and pilot tailorability. Pilot tailorability is an important requirement for both the input and output voice systems. The pilot should be able to tailor the voice input/output system to: specified words, specified voice quality (authoritative, suggestive), and determine which items under which conditions should be voice controllable (dynamic task allocation). For this dialogue to work intelligently, as intended, the pilot and PA must have a shared knowledge context.

As actual aircraft incorporate voice interactive systems, pilots will develop jargon, nicknames, abbreviations, and acronyms; an intelligent voice system must be able to accomodate such changes.

Table A-1

Vocabulary Listing

Mission phase: pre-takeoff

Note: Checklists may be accomplished by the system, monitored by the system while pilot accomplishes, or just displayed to the pilot. These options are pilot selectable.

Command	Meaning	Source	Response
preflight systems check BIT check		CNL	All systems go, except... (brief description)
start engines engine start		TI	engines started - normal
engine start checklist		CNL	(checklist displayed)
taxi check taxi checklist		TI	taxi checklist complete
takeoff check takeoff checklist		CNL	ready for takeoff safety check complete (flaps, engines, fuel, hydraulics, weapons)

Table A-1 (continued)

Vocabulary Listing

Mission phase: takeoff/climbout

Command	Meaning	Source	Response
abort checklist	stop takeoff	CNL	abort checklist complete (run appropriate emergency procedure checklist)
gear up		TI	
flaps up		CNL	
fence in fence check	prepare to enter enemy area	CNL TI TAC	fence check complete (weapon status)

Table A-1 (continued)

Vocabulary Listing

Mission phase: level off

Command	Meaning	Source	Response
level-off check	check fuel, timing, and critical systems	CNL	level-off checklist complete
vector to xxx (e.g., turnpoint 3)	course change	CNL	heading...(3 digit number) or turning to...(3 digit number); (after turn) steady...(3 digit number)

Table A-1 (continued)

Vocabulary Listing

Mission phase: descent for low-level

Command	Meaning	Source	Response
descent check low-level check	*check terrain following equipment*	CNL	*descent check complete;* altimeter xx.xx
go stealth max stealth passive	emitters off	CNL	emitters off

Table A-1 (continued)

Vocabulary Listing

Mission phase: low-level

Command	Meaning	Source	Response
auto tf	select terrain following mode	CNL	auto tf engaged (no command or response needed for manual)

Table A-1 (continued)

Vocabulary Listing

Mission phase: target ingress

Note: Weapons will not be fired by voice command, only prepared for firing. Manual arming and firing are mandatory for safety and pilot control.

Command	Meaning	Source	Response
air-to-ground	set cockpit for bomb delivery	CNL	(none needed; displays change)
kill xxx (target name)	prepare weapons to destroy named target	CNL	ready to kill xxx
primary target	from mission plan	CNL	ready (weapons set)
alternate secondary target	from mission plan	CNL	ready (weapons set)
new target xxx (e.g., airfield, factory, train, bridge)	re-mode weapons for new target	CNL	ready xxx (target name)
defend defense	use ecm, chaff, flares (as necessary to defend)	CNL	(status/count of countermeasures)
jam ecm chaff flares	turn on and expend as programmed	CNL	(display results)

Table A-1 (continued)

Vocabulary Listing

Mission phase: egress

Command	Meaning	Source	Response
bug out bug out xxx (direction) (e.g. east, west, 330)	depart enemy territory	TAC	(best route out of hostile area)
rtb pigeons homeplate homebase vector homeplate	go to home base	TAC CNL	(mission plan home)
bda battle check	battle damage	CNL	ok (or system(s) affected)
fence out	checklist for leaving hostile area	CNL	fence out complete
max range	best fuel consumption	CNL	(set up speed and altitude profile)
max speed	get out quickly	CNL	(set up speed and altitude profile)

Table A-1 (continued)

Vocabulary Listing

Mission phase: refueling

Command	*Meaning*	*Source*	*Response*
refueling checklist AR check		CNL	door open; ready for fuel
breakaway	emergency separation between tanker and fighter	CNL	(disconnect from tanker; check throttles at idle; if not, remind pilot)
post refueling checklist		CNL	door closed; fuel balanced

Table A-1 (continued)

Vocabulary Listing

Mission phase: descent and landing

Command	Meaning	Source	Response
descent check		CNL	descent check complete; altimeter xx.xx
xxx approach (e.g. TACAN, VORTAC, ILS, MLS)	approach type for planned base	CNL	(display requested approach)
runway xxx (used with approach type)	designate runway for landing	CNL	
gear down		TI	three green (put gear down)
flaps down		CNL	flaps down (put flaps down)
hook		TI	hook extended
go-around		CNL	(display and monitor go-around checklist)
emergency	support pilot execution; display check-lists; monitor actions	TI	(run appropriate checklist, if level of automation set to do so)

Table A-1 (continued)

Vocabulary Listing

Mission phase: shutdown

Command	Meaning	Source	Response
shutdown postflight mission complete	shutdown systems, including engines	CNL	shutdown complete (transfer actual mission data to data transfer module for debrief)

Table A-1 (continued)

Vocabulary Listing

Mission phase: air-to-air

Command	Meaning	Source	Response
air-to-air	mode cockpit	CNL	(cockpit mode change)
search	look for hostiles	CNL	(report findings)
passive search	look for hostiles with passive equipment only	CNL	(report findings)
monitor		CNL	(report change of intentions)
anchor CAP	orbit	TAC	(plan fuel, time, altitude)
bogey dope	info on unidentified aircraft	TAC CNL	(report bearing, range, altitude, aspect angle and type of aircraft, if known)
bandit dope bandit data	info on hostile aircraft	CNL	(same as above)
snap vector xxx (e.g. bandit #3) vector bogeys vector bandits vector fight intercept		CNL	(give vector requested)
long range short range heat radar go guns	weapons types	CNL	ready (display shoot cue)

Table A-1 (continued)

Vocabulary Listing

Mission phase: air-to-air

Command	Meaning	Source	Response
offense kill attack commit engage	go for the kill	CNL	(support pilot with tactics planning)
lock on radar lock lock bandits lock targets		CNL	locked
target xxx (e.g. MIG(s), fighters, heavy, chopper, trailer, leader) swap sort		CNL	(assign/designate target(s))
defense bugout hide evade avoid ECM chaff flares	defensive intentions commands	CNL	(report status of counter measures)

Table A-1 (continued)

Vocabulary Listing

Formation

Command	Meaning	Source	Response
where's two two's position		CNL	(report bearing, range, altitude aspect of wingman)
flight status formation status package status striker status		CNL	(report condition (fuel, systems problems) of formation, escortees)
data-link screen data-link to flight *tell the flight*		CNL	message sent (Note: Upon pilot acceptance of a new plan, that plan should automatically be sent to formation members)

Table A-1 (continued)

Vocabulary Listing

Tactics

Note: As currently used, the following terms can describe enemy actions as well as friendly tactics. The intended use here is to tell the Pilot's Associate the intentions of the pilot.

Command	Meaning	Source	Response
beam beaming beamer	plan to maneuver to 70-110 degrees aspect	TAC	(provide command steering for tactic)
blow through	plan no turn at merge	TAC	(provide command steering for tactic)
bracket	plan simal-taneous attack from opposite sides/altitudes	TAC	(provide command steering for tactic)
drag	target maneuvering to 60 degrees or less aspect	TAC	(provide command steering for tactic)
extend	gain energy and distance for re-engagement	TAC	(provide command steering for tactic)
flank flanking	target with a stable aspect 120-150 degrees	TAC	(provide command steering for tactic)

Table A-1 (continued)

Vocabulary Listing

Tactics

head-on	target with a stable aspect 160-180 degrees	TAC	(provide command steering for tactic)
high	target altitude 25,000-40,000	TAC	
medium	target altitude 10,000-25,000		
low	target altitude less than 10,000		

Table A-1 (continued)

Vocabulary Listing

Tactics

Command	Meaning	Source	Response
hook hooking xxx (left or right)	approach target from single side (left or right)	TAC	(provide command steering for tactic)
hot cold	geometry to pass in front (behind) target	TAC	(provide command steering for tactic)
pitchback xxx (left or right)	nose-high heading reversal (left or right)	TAC	(provide command steering for tactic)
press	continue attack	TAC	(provide command steering for tactic)
pince, pincer	(see bracket)	TAC	(provide command steering for tactic)
re-attack	attack again	TAC	(provide command steering for tactic)
scissors	a series of continuous turn reversals	TAC	(provide command steering for tactic)
shadow	follow indicated target, remain unobserved	TAC	(provide command steering for tactic)
shackle	one weave; single crossing of flight path	TAC	(provide command steering for tactic)

Table A-1 (continued)

Vocabulary Listing

Tactics

Command	Meaning	Source	Response
slice	nose-low heading reversal (left or right)	TAC	(provide command steering for tactic)
split, splitting	maneuver with separate targets	TAC	(provide command steering for tactic)
stern	intercept target at 6 o'clock	TAC	(provide command steering for tactic)
switch	change target	TAC	(provide command steering for tactic)

Table A-1 (continued)

Vocabulary Listing

Miscellaneous

Note 1: Commands are not all originally from CNL, but their use in a voice interactive cockpit was verified in the CNL study.

Note 2: System Response should be a brief, context-dependent reply.

Display Commands

zoom, close look, close up,
spotlight, expand, declutter,
give me, gimme, options, HUD,
God's eye, big picture, HSD,
friendlies

Sub-System Commands

engine(s), fuel, hydraulic(s),
flight control(s), sensor(s), weapons,
stores, ops check, autopilot, command
steering, radar, IRST, IR, countermeasures,
system(s), status, emergency, fix it, check,
checklist, auto, manual

Other Commands

radio(s), comm, HF, UHF, VHF, victor,
uniform, transmit, data-link, pass, relay,
notify, zap, transfer, send, tell

nav, navigation, VOR, ILS, TACAN, VFR,
turnpoint, waypoint, alternate, initial, primary,
destination, route, re-route, best route, safe
route, max range, min fuel, safe passage
min time, new route, (geographic names in
region: base names, cities, etc.)

Table A-1 (concluded)

Vocabulary Listing

Miscellaneous

Other Commands

IFF/SIF, mode 1, 2, 3, 4, squawk

who, what, why, when, where, how

louder, softer, bright(er), dim(mer),
off, on, standby, say again, cancel

zero..nine, ten..hundred, thousand,
alpha..zulu (A - Z phonetic)

accept, roger, wilco, yes, acknowledge
negative, no
o'clock, high, low, above, below
North, South, East, West
up, down, left, right, center
(tactical call signs: eagle, shark, sabre,,,)

APPENDIX B

ANALYZE: Grammar Analysis

Description

The ANALYZE program is adapted from McKeeman et al (1970) and has two purposes. The first is to assist in analyzing and debugging BNF grammars. The second is to prepare a file for inclusion in subsequent Pascal programs. This file is a set of table declarations which embody the parsing decision rules.

Input

Input to ANALYZE is a file named ANALYZE.BNF. It contains a grammar expressed in BNF; an example is given in Table B-1. This sample grammar was created for illustrative purposes only and is not meant to reflect any "real world" considerations. With the single exception of the special character string "EOS" which is used to mark the end of a sentence, all character strings in the grammar begin and end with a "<" and a ">" respectively. Each line in the grammar is a rule called a production (going from left to right during analysis) or reduction (going from right to left during synthesis).

As shown in Table B-1, asterisks are used ("<*" and "*>") to highlight strings which are terminals. Terminals never appear on the left side of a rule; non-terminals always appear somewhere on the left side of a rule. Expressed as a parse tree, terminals are terminal nodes and non-terminals are interior nodes Terminals are used as "part-of-speech" entries in the lexicon.

If a rule begins with one or more blanks, then its left side is taken to be the left side of the previous rule. For example, line number three is five blanks followed by <VERB-PHR> <NOUN-PHR>. It is interpreted as <SENTENCE> <VERB-PHR> <NOUN-PHR>. Thus, there are two definitions for <SENTENCE>. There can be as many as six entries per line.

Any word processor or text editor may be used to prepare a grammar file, so long as the resulting file contains no "funny" characters (e.g., control characters in a word processing document). An asterisk in column one means that the line is to be treated as a comment. Once this file has been prepared, simply type ANALYZE to execute the program.

Output

Output from ANALYZE is quite extensive In addition to messages which appear on the screen, ANALYZE produces two files, ANALYZE.ANS and ANALYZE.DEC

ANALYZE.ANS is a print file containing detailed analyses of the grammar contained in ANALYZE.BNF. If the grammar is fairly long, i.e., more than 100 rules, then ANALYZE.ANS will probably be more than 100K bytes. ANALYZE.DEC is a source code file which contains PASCAL declarations for the parsing decision tables used by subsequent programs.

Table B-2 presents the CRT screen after ANALYZE has been run using the file given in Table B-1. These data are presented because the run time for ANALYZE grows exponentially as a function of the number of rules in the grammar. If the run is taking several hours, which will happen when the number of rules approaches or exceeds 100, it is convenient to have information on the screen indicating that progress is being made. The terminology on the screen, such as F11, C1, and C2, is explained in the following sections.

Tables B-3 through B-8 show the contents of ANALYZE.ANS when the grammar in Table B-1 is analyzed. The following explanations are brief, serving only to indicate the nature of the information provided. For a detailed explanation of the underlying theory, the reader is referred to McKeeman et al (1970).

Table B-3 echoes the input grammar. For purposes of clarity and ease of reading, the reduction/production symbol ":: =" and the Logical Or symbol "|" are inserted. The numbers in the left column are the subscripts used in subsequent programs to refer to productions.

Table B-4 is the sorted BNF vocabulary. Non-terminals are in the left column and terminals in the right column. Within these two groups, the symbols are ordered first by length and second by collating sequence. For example, non-terminals 22 through 28 (<ADJ-LIST> through <VERB-PHR>) are a sub-group of length 10, within which they are ordered alphabetically. The symbols EOF and EOS signify "End Of File" and "End Of Sentence," respectively (EOF is automatically inserted by ANALYZE). They have special significance in the parsing programs. Again, the numbers in the left column are subscripts used to refer to BNF vocabulary symbols.

Table B-5 shows which symbols may appear as the heads of symbol strings. The symbol <VERB-TYPE> can serve as an example; it appears on line 31 in the table. This line shows a "Y" in columns 7, 8, 9, 15, 31, and 33. This means that it is permissible (according to the grammar rules) for <VERB-TYPE> to be the head symbol in the following strings:

<*VERB-ARM*>

<*VERB-AUX*>

<*VERB-CRT*>

<*VERB-SWITCH*>

< VERB-TYPE >

< VERB-SIMPLE >

These head relations correspond to productions 32 through 36 of the grammar in Table B-3.

After the head table has been computed, ANALYZE proceeds to generate the canonical sentential forms. These forms are all possible legal combinations of non-terminals and terminals. The recursion depth of this process is the same as the depth of the parse tree. As indicated at the bottom of Table B-5, there are 360 canonical sentential forms. F11 is a function, the value of which is the number of unique symbol triples in the sentential forms. In this case, it is 287.

Table B-6 presents the table for a function called the "stacking decision predicate," also known as C1. The rows in the table are possible contents of the top of the parse stack; this is the same as the BNF vocabulary. The columns are the possible terminals which can appear as tokens. The cell located by the intersection of a row and a column indicates what the parser should do, given this particular combination of stack top and token. The table entries are defined as:

blank: illegal combination

Y: yes, stack the token

N: no, do not stack (implies reduction pending)

#: conflict, need triple to resolve

For the grammar used in this example, symbol pairs are sufficient to resolve all syntactical ambiguities; no symbol triples are necessary. More sophisticated grammars will require triples.

Table B-7 lists productions which cannot be distinguished because their right parts are either identical or that one right part is a subset of the other. There are a number of tests which can determine how such productions are to be distinguished. As shown in the table, the most common one is length. Others are based on context:

(0, 1) or (1, 0)

(0, 1)

(1, 0)

(1, 1)

where (m, n) refers to "m" symbols in the stack and "n' symbols in the input text. Any failures to distinguish productions will be reported in the ANALYZE.ANS listing.

Table B-8 extends the conditions described in Table B-7. The extensions cause similar productions to be arranged into the order in which the context checks are to be made. The information presented in Tables B-7 and B-8 reflects the table look-up procedures employed by the parsing programs.

Table B-1

Contents of ANALYZE.BNF

A Test Grammar

```
<INPUT> <SENTENCE> EOS
<SENTENCE> <VERB-PHR>
        <VERB-PHR> <NOUN-PHR>
<NOUN-PHR> <NOUN-LIST>
        <*ARTICLE*> <NOUN-LIST>
        <ADJ-LIST> <NOUN-LIST>
        <*ARTICLE*> <ADJ-LIST> <NOUN-LIST>
<VERB-PHR> <VERB-TYPE>
        <ADV-LIST> <VERB-TYPE>
<ADJ-LIST> <ADJ-TYPE>
        <ADJ-LIST> <ADJ-TYPE>
<ADJ-TYPE> <*ADJ-SIZE*>
        <*ADJ-DISTANCE*>
        <*ADJ-SPEED*>
        <*ADJ-APPROX*>
        <*ADJ-SIT-AIR*>
<NOUN-LIST> <NOUN-TYPE>
        <NOUN-LIST> <NOUN-TYPE>
<NOUN-TYPE> <NOUN-STORES>
        <NOUN-WEAPONS>
        <NOUN-AIRPLANES>
<NOUN-STORES> <*NOUN-FUEL*>
        <*NOUN-STORES-OTH*>
<NOUN-WEAPONS> <*NOUN-OWN-WEAP*>
        <*NOUN-FOE-WEAP*>
<NOUN-AIRPLANES> <*NOUN-FIGHTER*>
        <*NOUN-BOMBER*>
<ADV-LIST> <ADV-TYPE>
        <ADV-LIST> <ADV-TYPE>
<ADV-TYPE> <*ADV-DIR*>
        <*ADV-TIME*>
<VERB-TYPE> <VERB-SIMPLE>
        <*VERB-AUX*> <VERB-SIMPLE>
<VERB-SIMPLE> <*VERB-SWITCH*>
        <*VERB-CRT*>
        <*VERB-ARM*>
```

Table B-2

Screen Messages from ANALYZE

```
C > analyze
Read Grammar
Sort Vocabulary
Compute Head Table
Sentential Productions
Index F11
Sort Productions
Compute C1
pairs
triples
conflicts
Compute C2
Production Choice Function
Print PASCAL Declarations
# Triples = 0 (max = 2000)
# F11 = 287 (max = 32000)
# Sentential forms =      360.
# Iterations = 1
You must rerun VOCAB
Time on:  10:39:15
Time off: 10:39:37
```

Table B-3

From ANALYZE.ANS

Input Grammar

```
1  <INPUT> ::= <SENTENCE> EOS
2  <SENTENCE> ::= <VERB-PHR>
3          | <VERB-PHR> <NOUN-PHR>
4  <NOUN-PHR> ::= <NOUN-LIST>
5          | <*ARTICLE*> <NOUN-LIST>
6          | <ADJ-LIST> <NOUN-LIST>
7          | <*ARTICLE*> <ADJ-LIST> <NOUN-LIST>
8  <VERB-PHR> ::= <VERB-TYPE>
9          | <ADV-LIST> <VERB-TYPE>
10 <ADJ-LIST> ::= <ADJ-TYPE>
11         | <ADJ-LIST> <ADJ-TYPE>
12 <ADJ-TYPE> ::= <*ADJ-SIZE*>
13         | <*ADJ-DISTANCE*>
14         | <*ADJ-SPEED*>
15         | <*ADJ-APPROX*>
16         | <*ADJ-SIT-AIR*>
17 <NOUN-LIST> ::= <NOUN-TYPE>
18         | <NOUN-LIST> <NOUN-TYPE>
19 <NOUN-TYPE> ::= <NOUN-STORES>
20         | <NOUN-WEAPONS>
21         | <NOUN-AIRPLANES>
22 <NOUN-STORES> ::= <*NOUN-FUEL*>
23          | <*NOUN-STORES-OTH*>
24 <NOUN-WEAPONS> ::= <*NOUN-OWN-WEAP*>
25          | <*NOUN-FOE-WEAP*>
26 <NOUN-AIRPLANES> ::= <*NOUN-FIGHTER*>
27           | <*NOUN-BOMBER*>
28 <ADV-LIST> ::= <ADV-TYPE>
29         | <ADV-LIST> <ADV-TYPE>
30 <ADV-TYPE> ::= <*ADV-DIR*>
31         | <*ADV-TIME*>
32 <VERB-TYPE> ::= <VERB-SIMPLE>
33         | <*VERB-AUX*> <VERB-SIMPLE>
34 <VERB-SIMPLE> ::= <*VERB-SWITCH*>
35          | <*VERB-CRT*>
36          | <*VERB-ARM*>
```

Table B-4

From ANALYZE.ANS

Sorted BNF Vocabulary

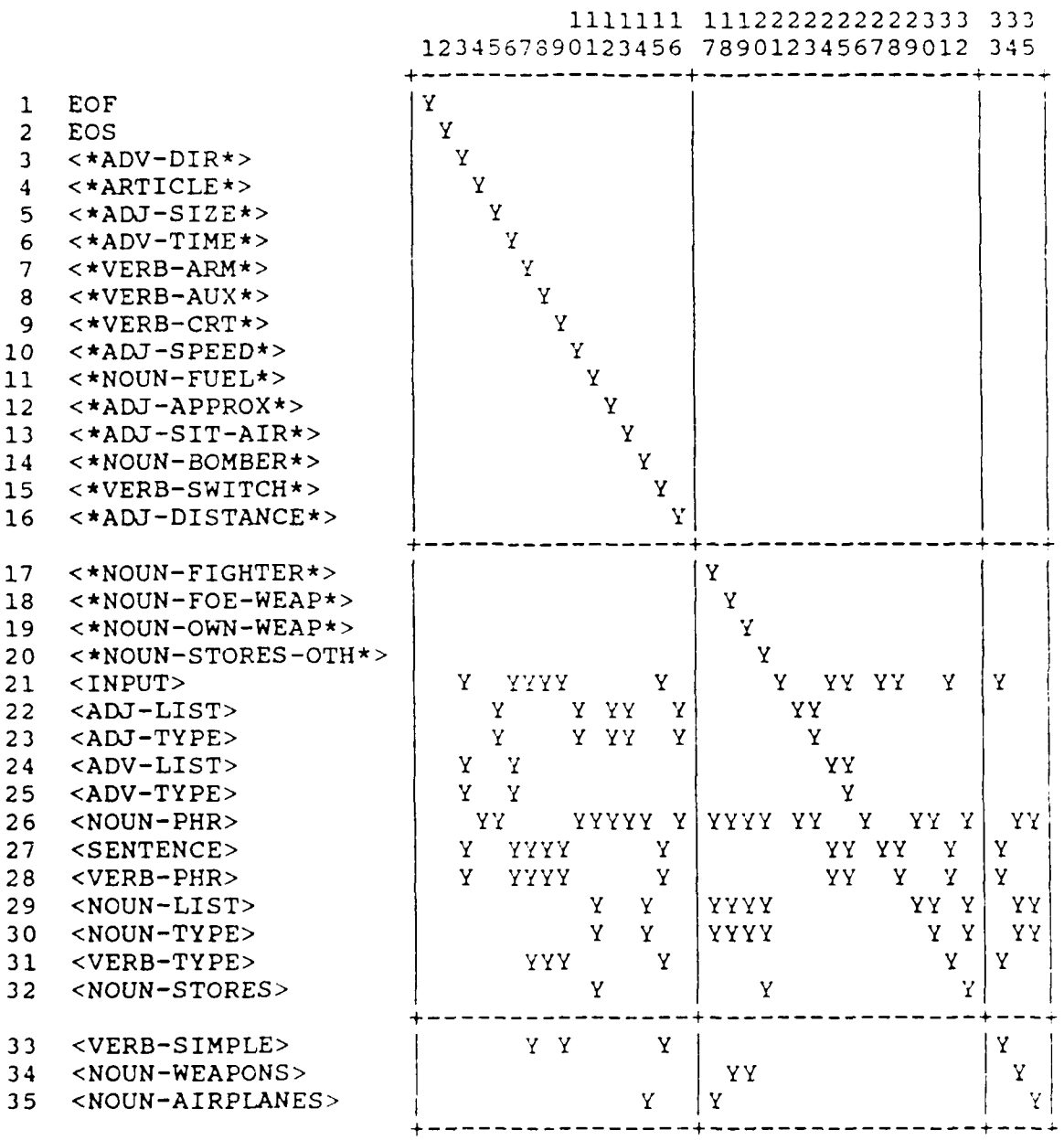
Non-Terminals 15	Terminals 20
1	EOF
2	EOS
3	<*ADV-DIR*>
4	<*ARTICLE*>
5	<*ADJ-SIZE*>
6	<*ADV-TIME*>
7	<*VERB-ARM*>
8	<*VERB-AUX*>
9	<*VERB-CRT*>
10	<*ADJ-SPEED*>
11	<*NOUN-FUEL*>
12	<*ADJ-APPROX*>
13	<*ADJ-SIT-AIR*>
14	<*NOUN-BOMBER*>
15	<*VERB-SWITCH*>
16	<*ADJ-DISTANCE*>
17	<*NOUN-FIGHTER*>
18	<*NOUN-FOE-WEAP*>
19	<*NOUN-OWN-WEAP*>
20	<*NOUN-STORES-OTH*>
21	<INPUT>
22	<ADJ-LIST>
23	<ADJ-TYPE>
24	<ADV-LIST>
25	<ADV-TYPE>
26	<NOUN-PHR>
27	<SENTENCE>
28	<VERB-PHR>
29	<NOUN-LIST>
30	<NOUN-TYPE>
31	<VERB-TYPE>
32	<NOUN-STORES>
33	<VERB-SIMPLE>
34	<NOUN-WEAPONS>
35	<NOUN-AIRPLANES>

Goal Symbol is: <INPUT>

Table B-5

From ANALYZE.ANS

Produced Head Symbols



Sentential productions

Maximum recursion depth 7
360. canonical sentential forms
F11 has 287 elements

Table B-6

From ANALYZE.ANS

C1 matrix for stacking decision

		1 2 3 4 5 6 7 8 9 0	1 1 1 1 1 1 1 1 2 3 4 5 6	1 1 1 2 7 8 9 0
1	EOF	Y	YYYY Y	
2	EOS	N		
3	<*ADV-DIR*>	N	NNNN N	
4	<*ARTICLE*>	Y	YYYYY Y	YYYY
5	<*ADJ-SIZE*>	N	NNNNN N	NNNN
6	<*ADV-TIME*>	N	NNNN N	
7	<*VERB-ARM*>	N NN	NNNNN N	NNNN
8	<*VERB-AUX*>	Y Y	Y	
9	<*VERB-CRT*>	N NN	NNNNN N	NNNN
10	<*ADJ-SPEED*>	N	NNNNN N	NNNN
11	<*NOUN-FUEL*>	N	N N	NNNN
12	<*ADJ-APPROX*>	N	NNNNN N	NNNN
13	<*ADJ-SIT-AIR*>	N	NNNNN N	NNNN
14	<*NOUN-BOMBER*>	N	N N	NNNN
15	<*VERB-SWITCH*>	N NN	NNNNN N	NNNN
16	<*ADJ-DISTANCE*>	N	NNNNN N	NNNN
17	<*NOUN-FIGHTER*>	N	N N	NNNN
18	<*NOUN-FOE-WEAP*>	N	N N	NNNN
19	<*NOUN-OWN-WEAP*>	N	N N	NNNN
20	<*NOUN-STORES-OTH*>	N	N N	NNNN
21	<INPUT>	N		
22	<ADJ-LIST>	Y	YYYYY Y	YYYY
23	<ADJ-TYPE>	N	NNNNN N	NNNN
24	<ADV-LIST>	Y	YYYY Y	
25	<ADV-TYPE>	N	NNNN N	
26	<NOUN-PHR>	N		
27	<SENTENCE>	Y		
28	<VERB-PHR>	N YY	YYYYY Y	YYYY
29	<NOUN-LIST>	N	Y Y	YYYY
30	<NOUN-TYPE>	N	N N	NNNN
31	<VERB-TYPE>	N NN	NNNNN N	NNNN
32	<NOUN-STORES>	N	N N	NNNN
33	<VERB-SIMPLE>	N NN	NNNNN N	NNNN
34	<NOUN-WEAPONS>	N	N N	NNNN
35	<NOUN-AIRPLANES>	N	N N	NNNN

Table entries summary
420
56 Y
224 N
0 #

No triples required

Table B-7

From ANALYZE.ANS

Context Checks for Equal and Embedded Right Parts

There are 22 and 22 valid contexts, respectively, for
11 <ADJ-LIST> ::= <ADJ-LIST> <ADJ-TYPE>
10 <ADJ-LIST> ::= <ADJ-TYPE>
They can be resolved by length

There are 6 and 6 valid contexts, respectively, for
29 <ADV-LIST> ::= <ADV-LIST> <ADV-TYPE>
28 <ADV-LIST> ::= <ADV-TYPE>
They can be resolved by length

There are 1 and 1 valid contexts, respectively, for
7 <NOUN-PHR> ::= <*ARTICLE*> <ADJ-LIST> <NOUN-LIST>
6 <NOUN-PHR> ::= <ADJ-LIST> <NOUN-LIST>
They can be resolved by length

There are 1 and 1 valid contexts, respectively, for
7 <NOUN-PHR> ::= <*ARTICLE*> <ADJ-LIST> <NOUN-LIST>
4 <NOUN-PHR> ::= <NOUN-LIST>
They can be resolved by length

There are 1 and 1 valid contexts, respectively, for
5 <NOUN-PHR> ::= <*ARTICLE*> <NOUN-LIST>
4 <NOUN-PHR> ::= <NOUN-LIST>
They can be resolved by length

There are 1 and 1 valid contexts, respectively, for
6 <NOUN-PHR> ::= <ADJ-LIST> <NOUN-LIST>
4 <NOUN-PHR> ::= <NOUN-LIST>
They can be resolved by length

There are 21 and 21 valid contexts, respectively, for
18 <NOUN-LIST> ::= <NOUN-LIST> <NOUN-TYPE>
17 <NOUN-LIST> ::= <NOUN-TYPE>
They can be resolved by length

Table B-7 (concluded)

From ANALYZE.ANS

Context Checks for Equal and Embedded Right Parts

```
There are  13 and  13 valid contexts, respectively, for
      9  <VERB-PHR> ::= <ADV-LIST> <VERB-TYPE>
      8  <VERB-PHR> ::= <VERB-TYPE>
      They can be resolved by length

There are  26 and  26 valid contexts, respectively, for
      33  <VERB-TYPE> ::= <*VERB-AUX*> <VERB-SIMPLE>
      32  <VERB-TYPE> ::= <VERB-SIMPLE>
      They can be resolved by length
```

Table B-8

From ANALYZE.ANS

C2 Production Choice Function

EOS as stack top will cause productions to be checked in this order:

1 <INPUT> ::= <SENTENCE> EOS
There will be no context check

<*ADV-DIR*> as stack top will cause productions to be checked in this order:

30 <ADV-TYPE> ::= <*ADV-DIR*>
There will be no context check

<*ADJ-SIZE*> as stack top will cause productions to be checked in this order:

12 <ADJ-TYPE> ::= <*ADJ-SIZE*>
There will be no context check

<*ADV-TIME*> as stack top will cause productions to be checked in this order:

31 <ADV-TYPE> ::= <*ADV-TIME*>
There will be no context check

<*VERB-ARM*> as stack top will cause productions to be checked in this order:

36 <VERB-SIMPLE> ::= <*VERB-ARM*>
There will be no context check

<*VERB-CRT*> as stack top will cause productions to be checked in this order:

35 <VERB-SIMPLE> ::= <*VERB-CRT*>
There will be no context check

<*ADJ-SPEED*> as stack top will cause productions to be checked in this order:

14 <ADJ-TYPE> ::= <*ADJ-SPEED*>
There will be no context check

Table B-8 (continued)

From ANALYZE.ANS

C2 Production Choice Function

<*NOUN-FUEL*> as stack top will cause productions to be checked in this order:

22 <NOUN-STORES> ::= <*NOUN-FUEL*>
There will be no context check

<*ADJ-APPROX*> as stack top will cause productions to be checked in this order:

15 <ADJ-TYPE> ::= <*ADJ-APPROX*>
There will be no context check

<*ADJ-SIT-AIR*> as stack top will cause productions to be checked in this order:

16 <ADJ-TYPE> ::= <*ADJ-SIT-AIR*>
There will be no context check

<*NOUN-BOMBER*> as stack top will cause productions to be checked in this order:

27 <NOUN-AIRPLANES> ::= <*NOUN-BOMBER*>
There will be no context check

<*VERB-SWITCH*> as stack top will cause productions to be checked in this order:

34 <VERB-SIMPLE> ::= <*VERB-SWITCH*>
There will be no context check

<*ADJ-DISTANCE*> as stack top will cause productions to be checked in this order:

13 <ADJ-TYPE> ::= <*ADJ-DISTANCE*>
There will be no context check

<*NOUN-FIGHTER*> as stack top will cause productions to be checked in this order:

26 <NOUN-AIRPLANES> ::= <*NOUN-FIGHTER*>
There will be no context check

Table B-8 (continued)

From ANALYZE.ANS

C2 Production Function

<*NOUN-FOE-WEAP*> as stack top will cause productions to be checked in this order:

25 <NOUN-WEAPONS> ::= <*NOUN-FOE-WEAP*>
There will be no context check

<*NOUN-OWN-WEAP*> as stack top will cause productions to be checked in this order:

24 <NOUN-WEAPONS> ::= <*NOUN-OWN-WEAP*>
There will be no context check

<*NOUN-STORES-OTH*> as stack top will cause productions to be checked in this order:

23 <NOUN-STORES> ::= <*NOUN-STORES-OTH*>
There will be no context check

<ADJ-TYPE> as stack top will cause productions to be checked in this order:

11 <ADJ-LIST> ::= <ADJ-LIST> <ADJ-TYPE>
There will be no context check
10 <ADJ-LIST> ::= <ADJ-TYPE>
There will be no context check

<ADV-TYPE> as stack top will cause productions to be checked in this order:

29 <ADV-LIST> ::= <ADV-LIST> <ADV-TYPE>
There will be no context check
28 <ADV-LIST> ::= <ADV-TYPE>
There will be no context check

<NOUN-PHR> as stack top will cause productions to be checked in this order:

3 <SENTENCE> ::= <VERB-PHR> <NOUN-PHR>
There will be no context check

Table B-8 (continued)

From ANALYZE.ANS

C2 Production Function

<VERB-PHR> as stack top will cause productions to be checked in this order:

2 <SENTENCE> ::= <VERB-PHR>
There will be no context check

<NOUN-LIST> as stack top will cause productions to be checked in this order:

7 <NOUN-PHR> ::= <*ARTICLE*> <ADJ-LIST> <NOUN-LIST>
There will be no context check
5 <NOUN-PHR> ::= <*ARTICLE*> <NOUN-LIST>
There will be no context check
6 <NOUN-PHR> ::= <ADJ-LIST> <NOUN-LIST>
There will be no context check
4 <NOUN-PHR> ::= <NOUN-LIST>
There will be no context check

<NOUN-TYPE> as stack top will cause productions to be checked in this order:

18 <NOUN-LIST> ::= <NOUN-LIST> <NOUN-TYPE>
There will be no context check
17 <NOUN-LIST> ::= <NOUN-TYPE>
There will be no context check

<VERB-TYPE> as stack top will cause productions to be checked in this order:

9 <VERB-PHR> ::= <ADV-LIST> <VERB-TYPE>
There will be no context check
8 <VERB-PHR> ::= <VERB-TYPE>
There will be no context check

<NOUN-STORES> as stack top will cause productions to be checked in this order:

19 <NOUN-TYPE> ::= <NOUN-STORES>
There will be no context check

Table B-8 (concluded)

From ANALYZE.ANS

C2 Production Choice Function

<VERB-SIMPLE> as stack top will cause productions to be checked in this order:

33 <VERB-TYPE> ::= <*VERB-AUX*> <VERB-SIMPLE>
There will be no context check
32 <VERB-TYPE> ::= <VERB-SIMPLE>
There will be no context check

<NOUN-WEAPONS> as stack top will cause productions to be checked in this order:

20 <NOUN-TYPE> ::= <NOUN-WEAPONS>
There will be no context check

<NOUN-AIRPLANES> as stack top will cause productions to be checked in this order:

21 <NOUN-TYPE> ::= <NOUN-AIRPLANES>
There will be no context check

Analysis complete for iteration 1
No errors detected

APPENDIX C

VOCAB: Lexicon Preparation

Description

VOCAB is used to create a lexicon file. Terminal symbols are defined by the grammar, such as the sample given in Table B-1. These terminal symbols are to be used as a kind of "part-of-speech" in the vocabulary definition file (described later). The first thing VOCAB does is ensure that all terminal symbols are used at least once as a part-of-speech in the vocabulary definition file. The second thing is to ensure that each part-of-speech in the vocabulary definition file is defined as a terminal symbol in the grammar. The last thing VOCAB does is to prepare the lexicon itself.

Input

Table C-1 shows sample input for VOCAB. It is a file named VOCAB.DEF which may be prepared with any suitable word processor. A maximum of four terminal symbols may be used for each word. Entries are free format, being separated by one or more blanks. Although the file is shown in alphabetical order, this is simply a matter of user convenience rather than a requirement. VOCAB also uses ANALYZE.DEC as the source for the terminal symbols.

Output

VOCAB will log its progress on the screen. If a terminal symbol is defined but not used as a part-of-speech or if a part-of-speech is used but not defined as a terminal symbol, the error is reported on the screen. Whether or not any errors are found, VOCAB will create a file named VOCAB.LEX. It contains the vocabulary sorted by collating sequence within length (all words of the same length are grouped together in alphabetical order). Each word is followed by four subscript values representing pointers into the terminals portion of the BNF vocabulary as given in Table B-4. In the case of the current sample grammar, these subscripts range from 3 through 20 (a zero means no entry).

The last entries in the lexicon VOCAB.LEX are a subscript table for the binary table look-up routines in PARSE and other programs. Suppose a word from a command is to be located in the lexicon and that its length is L. If the table is entered with L - 1, the value returned is a pointer into the vocabulary and parts-of-speech tables. It points to the first word of the group whose words are length L. For large vocabularies, this accomplishes an enormous savings in search time.

An abbreviated version of VOCAB.LEX is presented in Table C-2. For example, the word "ARM" has two (out of four) parts of speech assign to it: 7, which corresponds to <*VERB-ARM*> in Table B-4; and 19, which corresponds to <*NOUN-OWN-WEAP*>

in Table B-4. The last three lines in Table C-2 comprise the word-length look-up table (subscript values 0 through 16).

VOCAB also creates a file named SEMANT.PRO. It contains a skeleton procedure for extracting the semantics of commands when a command has been successfully parsed. It is described in the section on the SEMANT program.

Table C-1

VOCAB.DEF: a sample lexicon, source

```
A <*ARTICLE*> <*LETTER*>
ACTIVATE <*VERB-SWITCH*> <*VERB-ARM*>
AIR-TO-AIR <*ADJ-SIT-AIR*> <*VERB-CMD-SA*>
AIRCRAFT <*NOUN-ANY-PLANE*>
ALL <*ADJ-APPROX*>
ARM <*VERB-ARM*> <*NOUN-OWN-WEAP*>
AUTOMATIC <*ADJ-FUN*> <*NOUN-FUN*>
B <*LETTER*> <*VERB-EXIST*>
BANDIT <*NOUN-BANDIT*>
BASE <*NOUN-NAV-AIR*> <*NOUN-NAV-GRND*>
BELOW <*PREP-LOC*>
BINGO <*NOUN-FUEL*>
BIT <*NOUN-HEALTH*>
BOGEYS <*NOUN-ANY-PLANE*>
CHAFF <*NOUN-OWN-WEAP*>
EAST <*ADV-DIR*> <*NOUN-DIR*>
FEBA <*NOUN-NAV-GRND*>
MISSILES <*NOUN-OWN-WEAP*> <*NOUN-FOE-WEAP*>
QUICK <*NOUN-RADIO*>
RANGE <*NOUN-DISTANCE*> <*ADJ-DISTANCE*>
SAM <*NOUN-SAM*>
SEVENTEEN <*NUMBER*>
SPLASHED <*ADJ-SIT-AIR*>
THREAT <*NOUN-AAA*> <*NOUN-SAM*> <*NOUN-BANDIT*>
TO <*PREP-OBJ*> <*NUMBER*>
VECTOR <*NOUN-NAV-AIR*> <*NOUN-NAV-GRND*> <*NOUN-GEOM*>
```

Table C-2

VOCAB.LEX: a sample lexicon, converted

```
ALL 12 0 0 0
ARM 7 19 0 0
MISSILES 19 18 0 0
AIR-TO-AIR 13 0 0 0
16
1  1  26  60  153  310  443  586  717  808  853
879  898  905  906  910  912
```

APPENDIX D

PARSE: Command Parsing

Description

PARSE was written to provide detailed analyses of what happens when a particular grammar is applied to a set of pilot commands.

Input

Four files for PARSE must be prepared ahead of time. ANALYZE.DEC is automatically created by ANALYZE and contains the parsing decision tables. VOCAB.LEX is created by VOCAB. EXPAND.INC is a file of table declarations which change contracted words, such as "I'M," into their expanded equivalents, such as "I AM." An example of this file is presented in Table D-1. It is not presented as an optimal method; it was a quick fix to accomodate transcriptions of actual pilot commands collected in the field for the CNL study presented in Lizza et al (1987).

The files just described are Pascal "include" files. Anytime a new grammar is submitted to ANALYZE, the file ANALYZE.DEC is erased and VOCAB must be run again to recreate it. Then PARSE, as well as any other program using ANALYZE.DEC and/or VOCAB.LEX, must be recompiled.

Commands to be analyzed by PARSE are contained in a file named PARSE.TXT. For purposes of illustration in the following pages, a single command was used:

arm all air-to-air missiles

Note that all the words in this command are in the vocabulary definition file given in Table C-1 (see also Table C-2). PARSE.TXT probably should not contain more than a dozen commands because the program output is rather lengthy.

Once the program is running, the user is presented with two options:

V = verbose output

T = terse output

The response consequences are described in the next section.

Output

Output is a print file named PARSE.ANS. If terse output was chosen, the results will look like those presented in Table D-2. The word "arm" has two parts-of-speech and the

word "missiles" also has two. This results in four possible commands represented by parts-of-speech. In all cases, PARSE appends EOS to the command.

Note that errors are reported when the word "arm" is interpreted to be a noun referring to a weapon (anti-radiation missile). This is because the test grammar requires that commands begin with a verb phrase. Note also that the word "missiles" can represent "own" or "foe" missiles. In the latter case, the command is syntactically correct but semantically wrong; the grammar must be revised.

A partial listing of verbose output is given in Table D-3. The information presented reflects the various steps the parser goes through in selecting appropriate productions to reduce the input stream. This information is more easily understood if one is willing to become knowledgeable in the underlying theory (McKeeman et al, 1970) and its implementation in PARSE.

Table D-1

Word Expansion Tables: input word

```
Const
   MaxQ = 20;

Var
   QWord: Array[1..MaxQ] of S24;

Value
   QWord[1]  := 'I''M';
   QWord[2]  := 'IT''S';
   QWord[3]  := '''EM';
   QWord[4]  := 'WE''LL';
   QWord[5]  := 'THAT''S';
   QWord[6]  := 'LET''S';
   QWord[7]  := 'GIMME';
   QWord[8]  := 'WELL';
   QWord[9]  := 'LETS';
   QWord[10] := 'WHERES';
   QWord[11] := 'WHATS';
   QWord[12] := 'WERE';
   QWord[13] := 'WE''RE';
   QWord[14] := 'ILL';
   QWord[15] := 'IM';
   QWord[16] := 'I''VE';
   QWord[17] := 'HOWS';
   QWord[18] := 'IVE';
   QWord[19] := 'YOUVE';
   QWord[20] := 'WANNA';
```

Table D-1 (continued)

Word Expansion Tables: first output string

```
Var
    QWord1: Array[1..MaxQ] of S24;

Value
    QWord1[1]  := 'I';
    QWord1[2]  := 'IT';
    QWord1[3]  := 'THEM';
    QWord1[4]  := 'WE';
    QWord1[5]  := 'THAT';
    QWord1[6]  := 'LET';
    QWord1[7]  := 'GIVE';
    QWord1[8]  := 'WE';
    QWord1[9]  := 'LET';
    QWord1[10] := 'WHERE';
    QWord1[11] := 'WHAT';
    QWord1[12] := 'WE';
    QWord1[13] := 'WE';
    QWord1[14] := 'I';
    QWord1[15] := 'I';
    QWord1[16] := 'I';
    QWord1[17] := 'HOW';
    QWord1[18] := 'I';
    QWord1[19] := 'YOU';
    QWord1[20] := 'WANT';
```

Table D-1 (concluded)

Word Expansion Tables: second output string

```
Var
    QWord2: Array[1..MaxQ] of S24;

    Value
    QWord2[1]  := 'AM';
    QWord2[2]  := 'IS';
    QWord2[3]  := Null;
    QWord2[4]  := 'WILL';
    QWord2[5]  := 'IS';
    QWord2[6]  := 'US';
    QWord2[7]  := 'ME';
    QWord2[8]  := 'WILL';
    QWord2[9]  := 'US';
    QWord2[10] := 'IS';
    QWord2[11] := 'IS';
    QWord2[12] := 'ARE';
    QWord2[13] := 'ARE';
    QWord2[14] := 'WILL';
    QWord2[15] := 'AM';
    QWord2[16] := 'HAVE';
    QWord2[17] := 'IS';
    QWord2[18] := 'HAVE';
    QWord2[19] := 'HAVE';
    QWord2[20] := 'TO';
```

Table D-2

Printed Output From PARSE, Terse

```
**************************************************
Reducing Sentence # 1

ARM ALL AIR-TO-AIR MISSILES EOS

<*VERB-ARM*> <*ADJ-APPROX*> <*ADJ-SIT-AIR*>
<*NOUN-OWN-WEAP*> EOS


36 <VERB-SIMPLE> <==  <*VERB-ARM*>
32 <VERB-TYPE> <==  <VERB-SIMPLE>
8 <VERB-PHR> <==  <VERB-TYPE>
15 <ADJ-TYPE> <==  <*ADJ-APPROX*>
10 <ADJ-LIST> <==  <ADJ-TYPE>
16 <ADJ-TYPE> <==  <*ADJ-SIT-AIR*>
11 <ADJ-LIST> <==  <ADJ-LIST> <ADJ-TYPE>
24 <NOUN-WEAPONS> <==  <*NOUN-OWN-WEAP*>
20 <NOUN-TYPE> <==  <NOUN-WEAPONS>
17 <NOUN-LIST> <==  <NOUN-TYPE>
6 <NOUN-PHR> <==  <ADJ-LIST> <NOUN-LIST>
3 <SENTENCE> <==  <VERB-PHR> <NOUN-PHR>
1 <INPUT> <==  <SENTENCE> EOS
```

Table D-2 (continued)

Printed Output From PARSE, Terse

```
**************************************************
Reducing Sentence # 2

ARM ALL AIR-TO-AIR MISSILES EOS

<*NOUN-OWN-WEAP*> <*ADJ-APPROX*> <*ADJ-SIT-AIR*>
<*NOUN-OWN-WEAP*> EOS


***ERROR*** illegal symbol pair ->   EOF   <*NOUN-OWN-WEAP*>
Stack || Token || Input Stream
EOF || <*NOUN-OWN-WEAP*> || ALL AIR-TO-AIR MISSILES EOS
```

Table D-2 (continued)

Printed Output From PARSE, Terse

```
**************************************************

Reducing Sentence # 3
ARM ALL AIR-TO-AIR MISSILES EOS

<*VERB-ARM*> <*ADJ-APPROX*> <*ADJ-SIT-AIR*>
<*NOUN-FOE-WEAP*> EOS


36 <VERB-SIMPLE> <==  <*VERB-ARM*>
32 <VERB-TYPE> <== <VERB-SIMPLE>
8 <VERB-PHR> <== <VERB-TYPE>
15 <ADJ-TYPE> <== <*ADJ-APPROX*>
10 <ADJ-LIST> <== <ADJ-TYPE>
16 <ADJ-TYPE> <== <*ADJ-SIT-AIR*>
11 <ADJ-LIST> <== <ADJ-LIST> <ADJ-TYPE>
25 <NOUN-WEAPONS> <== <*NOUN-FOE-WEAP*>
20 <NOUN-TYPE> <== <NOUN-WEAPONS>
17 <NOUN-LIST> <== <NOUN-TYPE>
6 <NOUN-PHR> <== <ADJ-LIST> <NOUN-LIST>
3 <SENTENCE> <== <VERB-PHR> <NOUN-PHR>
1 <INPUT> <== <SENTENCE> EOS
```

Table D-2 (concluded)

Printed Output From PARSE, Terse

```
**************************************************
Reducing Sentence # 4

ARM ALL AIR-TO-AIR MISSILES EOS

<*NOUN-OWN-WEAP*> <*ADJ-APPROX*> <*ADJ-SIT-AIR*>
<*NOUN-FOE-WEAP*> EOS


***ERROR*** illegal symbol pair ->   EOF  <*NOUN-OWN-WEAP*>
Stack || Token || Input Stream
EOF || <*NOUN-OWN-WEAP*> || ALL AIR-TO-AIR MISSILES EOS
```

Table D-3

Printed Output From PARSE, Verbose

```
**************************************************
Reducing Sentence # 1

ARM ALL AIR-TO-AIR MISSILES EOS

<*VERB-ARM*> <*ADJ-APPROX*> <*ADJ-SIT-AIR*>
<*NOUN-OWN-WEAP*> EOS

Stacking decision predicate for
EOF <*VERB-ARM*> = 1
Stacking = True
Stacking decision predicate for
<*VERB-ARM*> <*ADJ-APPROX*> = 2
Stacking = False
Do Reduction of
Stack || Token
|| Input Stream
EOF <*VERB-ARM*> || <*ADJ-APPROX*>
|| AIR-TO-AIR MISSILES EOS
Production 36: <VERB-SIMPLE> <= = <*VERB-ARM*>
has masked triple of 0 0 0 0
ProdOK   Production 36 has context case of 0
Triple matched and prod ok
Stack || Token
|| Input Stream
EOF <VERB-SIMPLE> || <*ADJ-APPROX*>
|| AIR-TO-AIR MISSILES EOS
Stacking decision predicate for
<VERB-SIMPLE> <*ADJ-APPROX*> = 2
Stacking = False
Do Reduction of
Stack || Token
|| Input Stream
EOF <VERB-SIMPLE> || <*ADJ-APPROX*>
|| AIR-TO-AIR MISSILES EOS
Production 33: <VERB-TYPE> <= = <*VERB-AUX*>
<VERB-SIMPLE>
has masked triple of 0 0 0 1
Production 32: <VERB-TYPE> <= = <VERB-SIMPLE>
...
(continues at great length)
...
```

APPENDIX E

SEMANT: Semantics Extraction

Description

SEMANT is meant to be a starting point for developing the semantic portion of speech understanding software. It is essentially PARSE with the printed output statements removed and the file SEMANT.PRO, described below, "included" in the source code.

Input

ANALYZE.DEC and EXPAND.INC are "include" files described in Appendix D. Commands to be analyzed by SEMANT should be put in a file named SEMANT.TXT.

Table E-1 presents a partial listing of SEMANT.PRO. It is a skeleton procedure named "Synthesize" which is meant to be fleshed out with instructions in each of the case statements. The array "SynthList" contains the numbers of the productions/reductions (NSynths of them) in the order in which they are to be applied. The array "TermList" contains, where appropriate, the subscript of the first terminal of the production P. The value of this subscript, which is the pointer to the appropriate entry in the lexicon, selects the appropriate case statement. The values printed are based on a 910 word lexicon (Lizza et al, 1987), not the file given in Table C-1.

Output

Table E-2 presents the output from SEMANT when processing the command "arm all air-to-air missiles." Underneath the input command are the subscript values of the locations of the words in the lexicon. The two columns of digits under the heading "success" are the production number and the terminal/word number (0 means no terminal in the production). Obviously this print-out will change significantly as code is added to SEMANT.PRO. Adding code to SEMANT.PRO, however, depends entirely on the assumptions, or the fidelity of the simulation, with respect to the mechanization of the cockpit system being studied. This mechanization will be reflected in the types of displays and functional capabilities that are presented to the pilot.

Using a data base of approximately 2000 pilot commands, this version of SEMANT processed commands at the rate of 10 per second on an IBM PC/AT. This rate includes the time necessary to read each command, one at a time, from disk.

Table E-1

Skeleton Procedure for Extracting Semantics

```
Procedure Synthesize;
  Var P: Integer;

  Procedure BadTop;
    Begin
      WriteIn(OutFile, '*** ERROR ***');
      WriteIn(OutFile, 'Production = ', SynthList[P]);
      WriteIn(OutFile, 'Stack Top = ', TermList[P]);
    end;

Begin
  For P := 1 to NSynths do
    Begin
      WriteIn(OutFile, SynthList[P]:4, TermList[P]:5);
        Case SynthList[P] of
          1: Begin
              (* <INPUT> <==> <SENTENCE> EOS *)
            end;
          2: Begin
              (* <SENTENCE> <== <VERB-PHR> *)
            end;
          3: Begin
              (* <SENTENCE> <== <VERB-PHR> <NOUN-PHR> *)
            end;
          4: Begin
              (* <NOUN-PHR> <== <NOUN-LIST> *)
            end;
          5: Begin
              (* <NOUN-PHR> <== <*ARTICLE*> <NOUN-LIST> *)
              Case TermList[P] of
                1: Begin  (* A *)
                     end;
                28: Begin  (* AN *)
                      end;
                141: Begin  (* THE *)
                       end;
              Otherwise BadTop;
              end;
            end;
```

Table E-1 (concluded)

Skeleton Procedure for Extracting Semantics

```
        6 : Begin
              (* <NOUN-PHR> <= = <ADJ-LIST> <NOUN-LIST> *)
            end;
              .
              .
              .
        (productions 7 through 35 would appear here)
              .
              .
              .
        36: Begin
              (* <VERB-SIMPLE> <= = <*VERB-ARM*> *)
              Case TermList[P] of
                718: Begin  (* ACTIVATE *)
                       end;
                 68: Begin  (* ARM *)
                       end;
                Otherwise BadTop;
              end;
            end;
      end;
    end;
end;
```

Table E-2

Printed Output From SEMANT

```
**************************************************
Reducing Sentence # 1

ARM ALL AIR-TO-AIR MISSILES EOS

68  63  856  761  82

<*VERB-ARM*> <*ADJ-APPROX*> <*ADJ-SIT-AIR*>
<*NOUN-OWN-WEAP*> EOS


Success
36    68
32     0
8      0
15    63
10     0
16   856
11     0
24   761
20     0
17     0
6      0
3      0
1     82
```

Table E-2 (continued)

Printed Output From SEMANT

```
**************************************************
Reducing Sentence # 2

ARM ALL AIR-TO-AIR MISSILES EOS

68  63  856  761  82
<*NOUN-OWN-WEAP*> <*ADJ-APPROX*> <*ADJ-SIT-AIR*>
<*NOUN-OWN-WEAP*> EOS


***ERROR*** illegal symbol pair ->   EOF  <*NOUN-OWN-WEAP*>

EOF || <*NOUN-OWN-WEAP*> || ALL AIR-TO-AIR MISSILES EOS
1 || 18


Failure
```

Table E-2 (continued)

Printed Output From SEMANT

```
**************************************************
Reducing Sentence # 3

ARM ALL AIR-TO-AIR MISSILES EOS

68  63  856  761  82

<*VERB-ARM*> <*ADJ-APPROX*> <*ADJ-SIT-AIR*>
<*NOUN-FOE-WEAP*> EOS


Success
36    68
32     0
8      0
15    63
10     0
16   856
11     0
25   761
20     0
17     0
6      0
3      0
1     82
```

END

DATE

FILMED

DTIC

JULY 88