(2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

TACTICAL EVIDENTIAL REASONING:
AN APPLICATION OF
THE DEMPSTER-SHAFER THEORY OF EVIDENCE

by

William H. Cleckner, IV

September, 1985

Thesis Advisor: P. S. Fischbeck

Approved for public release; distribution is unlimited.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. ADA 162 178	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Tactical Evidential Reasoning: An Application of the Dempster-Shafer Theory of Evidence		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis September, 1985
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) William H. Cleckner, IV		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943-5100		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943-5100		12. REPORT DATE September, 1985
		13. NUMBER OF PAGES 139
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Evidential Reasoning, Dempster-Shafer, Decision Aid, Decision Support System, Data Fusion, Evidence Combination, Inference from Evidence, Command and Control, Situation Development Analysis, All Source Analysis System, ASAS

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

This thesis demonstrates the use of the Dempster-Shafer theory of evidence on the battlefield. The use of the theory in a Decision Aid or Decision Support System for the Intelligence Analyst will speed the force commander's Command and Control Cycle.

The Command and Control Cycle is modelled and a benefit of enhanced command and control is described. The Dempster-Shafer theory is discussed using tactical battlefield examples. A

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

20. Dempster-Shafer Decision Aid is presented as well as methods for improving computational speed. A specific application area, Situation Development Analysis in the All Source Analysis System (ASAS) is proposed.

Approved for public release; distribution is unlimited.

Tactical Evidential Reasoning: An Application of the Dempster-Shafer Theory of Evidence

by

William H. Cleckner, IV
Captain, United States Army
B.S., United States Military Academy, 1975

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1985

Author: William H Cleckner III
William H. Cleckner, IV

Approved by: P.S. Fischbeck
P.S. Fischbeck, Thesis Advisor

Jack B Gifford
J.B. Gifford, Second Reader

Alan R Washburn
A.R. Washburn, Chairman,
Department of Operations Research

K.T. Marshall
Kneale T. Marshall,
Dean of Information and Policy Sciences

ABSTRACT

This thesis demonstrates the use of the Dempster-Shafer theory of evidence on the battlefield. The use of the theory in a Decision Aid or Decision Support System for the Intelligence Analyst will speed the force commander's Command and Control Cycle.

The Command and Control Cycle is modelled and a benefit of enhanced command and control is described. The Dempster-Shafer theory is discussed using tactical battlefield examples. A Dempster-Shafer Decision Aid is presented as well as methods for improving computational speed. A specific application area, Situation Development Analysis in the All Source Analysis System (ASAS) is proposed.

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to thank the Command and Control Analysis Division, Studies and Analysis Directorate, of the Combined Arms Operations Research Activity (CAORA), Fort Leavenworth, Kansas. Their support during my Experience Tour kindled my interest in Decision Aids/Decision Support Systems.

I would also like to thank LCDR Paul Fischbeck, my thesis advisor, for his guidance and motivation. Without his help, this thesis could not have been written.

A special thanks goes to my wife, Pam, whose constant support helped me through my studies and thesis effort.

I. INTRODUCTION

A. BACKGROUND

The nature or battle is rapidly changing.

> The development of modern warfare is outstriping our ability to manage it effectively. The speed, power and sophistication of modern weapons and the political and geographical complexities of potential battlefields in places like Europe and the Middle East have placed the commander in a rapidly moving, data-rich environment. [Ref. 1: p. 50]

Many publication on the state of the modern battlefield use the same descriptive adjectives: intense, dynamic, mobile, rapidly changing, data-rich, etc. The quoted article goes on to stress that modern decision aids are needed to assist the battlefield decision maker ". . . by expanding his ability to rapidly and effectively analyze the data becoming available." [Ref. 1: p. 50] However, to assist the force commander on the battlefield, much military research has emphasized the need to help the field commander and staff in deciding a course of action in critical battlefield situations [Ref. 2].

B. THE PROBLEM

The commander's decisions on the battlefield are primarily influenced by three factors: the environment, the enemy force, and the friendly force. The only factor that the force commander can directly control is the friendly force. If this control is to be effective, the commander must have intelligence about the environment, and the enemy force. In particular, the commander in the Airland battlefield needs a rapid, accurate assessment of the current enemy situation and the enemy's intentions.

The commander's information requirements place an ever increasing responsibility on the intelligence analyst. Raw data from battlefield sensors must be converted to intelligence as rapidly as possible. Without a proper understanding of the enemy's capabilities and intentions, the likelihood of selecting an appropriate course of action is extremely small.

C. THESIS OBJECTIVES

The primary objective of this thesis is to demonstrate the use of the Dempster-Shafer theory of evidence on the battlefield to help correlate data for predicting enemy intentions. Specifically, the thesis will:

1. Demonstrate the use of the Dempster-Shafer theory of evidence in a tactical military intelligence decision aid.
2. Create a decision aid prototype that uses the Dempster-Shafer theory.
3. Analyze techniques for reducing the computational complexity and calculation time required for Dempster-Shafer.

To understand the role of evidential reasoning on the battlefield, Chapter II of the thesis will discuss the tactical command and control process. It will also describe the problem of the intelligence analyst, unaided by an evidential decision aid, as he deals with the bombardment of battlefield data. Ennancement of the command and control cycle will be discussed, leading to an example showing the benefits of an improved cycle.

In Chapter III, the concept or evidential reasoning will be introduced. Two methods for evidential reasoning, Boolean Logic and Bayesian Inference, will be investigated to expose their shortcomings for use in battlefield

situations. The Dempster-Shafer theory of evidence, its advantages and disadvantages will be presented.

Chapter IV will describe the Dempster-Shafer theory decision aid created to assist the military intelligence analyst. The aid functions in a cyclical process, receiving each bit of data from the user, combining it with previous data, and presenting its conclusions to the user. Reduction in computational complexity and calculation time of the Dempster-Shafer method are also discussed. These computational techniques speed up the hierarchical summations required by the Dempster-Shafer theory, but they also reduce the scope of the problem. The reduction methods can be applied to evidential reasoning on the battlefield.

Chapter V discusses the battlefield intelligence analyst's job of Situation Development Analysis, a specific application area for the Dempster-Shafer decision aid. The enhancement of Situation Development Analysis by a Dempster-Shafer decision aid will lead to benefits of improved command and control.

A listing of the PASCAL code written for the decision aid in Chapter IV is contained in Appendix A.

II. BATTLEFIELD COMMAND AND CONTROL

The opportunity for battlefield applications of the Dempster-Shafer theory of evidence stems from the need to shorten a commander's command and control cycle. With a shorter cycle, a force commander can recognize enemy intentions and strike quickly enough to disrupt the enemy operation. With a quick reaction, the battlefield initiative can be seized before the opposing force can react to the new situation. The now weakened and off balanced enemy force will be easier to defeat. This process of converting data into action more quickly than the opponent is referred to as turning within the enemy's decision cycle.

Reducing the commander's command and control cycle time requires the capability to react more quickly than ever before. The faster and deeper that the friendly force can interdict the enemy's forces, the more successful the operation.

Enhanced command and control has emerged as a solution for dealing with a dynamic and data enriched battlefield while reducing the commander's reaction time. This chapter will discuss the command and control cycle, the data flow in the cycle, current enhancements to the cycle, and present an example that shows the benefit of reducing cycle time by enhancing command and control.

A. THE COMMAND AND CONTROL NETWORK

Command and control is an extremely complex battlefield function by which the commander and staff allocate resources, direct unit movement, and coordinate operations. Command and control can be modelled as a continuing cycle

within a network of nodes as illustrated in Figure 2.1. Within this cycle are processes, depicted as the nodes, that can be enhanced to reduce cycle time. These nodes have been described in many ways by many experts [Ref. 3], but will be designated here as Collection, Interpretation, Decision, and Action.

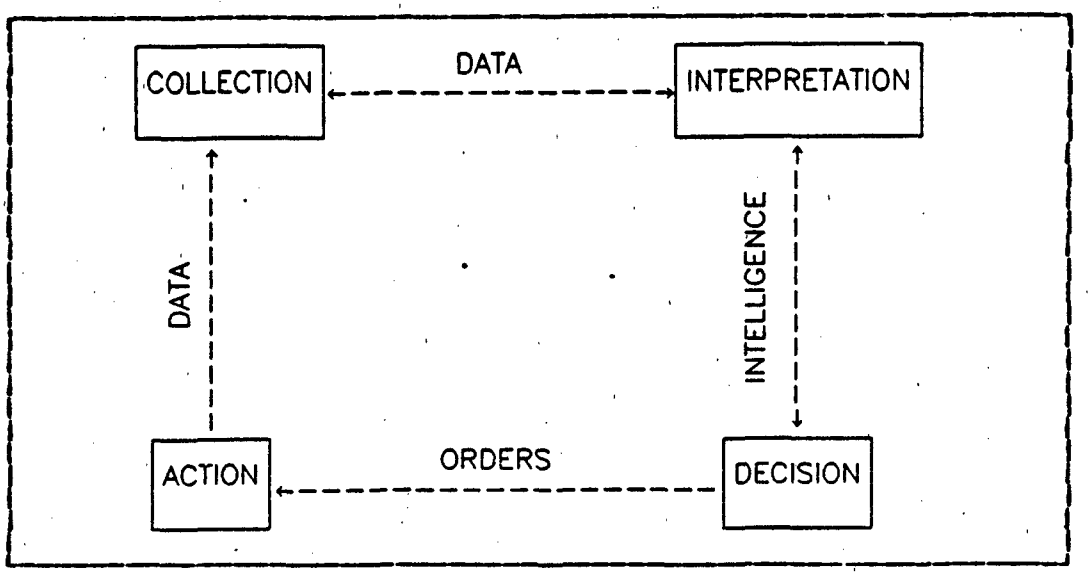

Figure 2.1 Command and Control Cycle Network.

To understand the command and control network, and the cycling within it during battle, it is necessary to investigate each node of the network. Within each node, functions that can be enhanced or eliminated may be discovered. The improvement of node efficiency will reduce cycle time and help in the effort to interdict the enemy force as early as possible.

The nodes of the cycle network depicted in Figure 2.1 are as follows.

1. Collection

The Collection node of this network describes the activity of gathering data about friendly and enemy forces by all means available. Miller and Cushman [Refs. 5,1], describe this activity as sensing, but sensing may include the recognition or interpretation of the data received. For purposes of this thesis, sensors only gather the data and do not modify it in any way.

Collection is a continuing process and does not realize divisions between battles or battlefields. A sensor may receive data that may has no effect in its area of operation. Yet, this same data may be of critical importance to the commander. Nonetheless, all of this data will be transmitted to the Interpretation node.

2. Interpretation

Interpretation is the combination, evaluation, and translation of raw data into intelligence. The result of this activity is an understanding of the battlefield situation including enemy intentions.

Interpretation, like Collection, is a continuing process. The intelligence analyst must deal with the continuous stream of relevant and irrelevant data arriving from the Collection node. Decision aids are used to help the intelligence analyst comprehend the entire battlefield picture presented by this onslaught of data.

The arc connecting the Collection and Interpretation node indicates flow in both directions. As the battlefield situation develops, sensors may be directed to change positions or sense other areas of the battlefield. Consequently, the connecting arc depicts coordination between these nodes.

3. Decision

After recognizing the enemy intentions, the friendly force must react. This node describes the commander's decision process that chooses the course of action which optimizes possibility of friendly success given the interpreted course of enemy action. This success can be measured by the minimization of friendly losses, the maximization of enemy losses, or by any measure of effectiveness or any measure of performance considered essential by the commander.

It is possible that the commander does not have enough information to justify a course of action. He may then request more data collection or direct surveillance in other areas. The commander will inform the Interpretation node of his Essential Elements of Information (EEI) or Priority Intelligence Requirements (PIR), those intelligence items of the utmost importance to the force. Therefore, the arc connecting the Interpretation and Decision nodes is bi-directional.

4. Action

The last node of the network represents the movement of troops into battle executing the course of action from the Decision node. This activity may range from a complicated maneuver to a reactionary tactic in a surprise encounter.

The arc from the Decision to Action node is unidirectional. This does not mean that subordinate commanders do not coordinate or respond to superiors, but it depicts the final orders of the higher-level commander after all planning is accomplished.

The command and control network ends in the flow of data from the Decision node to the Action node, see Figure 2.1, and therefore, can only function after the preceding nodes have completed their processes.

Of course, the enemy must also cycle through a similar network during the battle. During operations, each force must constantly collect, interpret, decide, and act at all levels of command within the battlefield. The force that executes their cycle the fastest will have the battlefield advantage.

B. ENHANCING THE NETWORK

There are many areas within the command and control network that can be enhanced by some type of automation. Stewart, Ross, and Tiede [Ref. 4] have identified functional areas in any organization where the human processor can be improved by introducing automation. Many of these areas have applications in the command and control network. Table 1 depicts these processes, the advantage gained by automation, and the areas of the command and control Network where they apply.

1. Enhancement Of Data Flow Through The Network

The arcs which connect the nodes of the command and control cycle network represent data flow. The Collection node passes data to the Interpretation node which passes data, or in this frame, intelligence to the Decision node. Here, possible courses of action are analyzed to select the one that best fulfills the commander's effectiveness measure. Next, the Decision node sends orders, which consist of the directions to units in the force, to the Action node. These force units will then execute the chosen course of action. The force units then act or interact with the enemy forces creating more data to be gathered by the Collection node and the cycle continues.

Much research has gone into the area of improving data handling and flow within the network [Ref. 5]. The

TABLE 1

AREAS FOR ENHANCEMENT BY AUTOMATION

Process	Applicable Network Node	Man Unaided By Machine	Machine Without Man	Man-Machine Interaction Leverage**
Receive/ Transmit	All	Communicates Mood of Sender	Increased Effective channel capacity; Hard Copy	Little
Interpret/ Validate	Interpretation	Only man can flesh out incomplete patterns and generate new hypotheses and tests for them	Can only extend human memory (associated operations) and facilitate hypotheses testing (calculation)	Tremendous
Evaluate/ Coordinate	Interpretation	Only man can interpret in context and generate hypotheses and insights	Can only extend human memory and facilitate coordination based on a priori rule	Significant
Project/ Extrapolate	Interpretation	Only man can define projection and extrapolation parameters; involves hypothesis generation	Can only extend memory and facilitates calculations	Tremendous

** Man-Machine Interaction Leverage is defined by the author as a measure of Potential Gain when man is aided by machine.

Tactical Operating System (TOS) and the Maneuver Control System (MCS) key on getting data to the commanders and presenting them with the best representation of the battlefield through improved communications capabilities.

Increased data flow is an intuitively appealing means of quickening the command and control cycle, but it also presents a variety of problems. In a fast moving, data rich battlefield, data will be flowing on the arc from the Collection node to the Interpretation node at an incredible rate [Ref. 1]. This flow will easily overwhelm the human processor at the heart of the Interpretation node.

In the Interpretation node, the data collected by all sensors available to the commander and staff (usually referred to as "all source") is transformed by humans into intelligence. Due to the high speed of input to the Interpretation node and the slow human interface, the input to the commander at the Decision node is only as fast as the intelligence analyst processing the data. In order to maintain the required speed, the analyst may tend to ignore or only briefly review input, losing a more complete battlefield picture. This problem suggests the need to reduce the research emphasis from systems that enhance data flow to systems that support the analyst's conversion of data into intelligence.

2. Enhancement Of The Network Processes

Inside each node of the command and control cycle data is being processed into a useful form for that particular node. If these processes inside the nodes are not streamlined, the command and control cycle will idle, waiting for a node to complete its activity. Enhancement of the network processes will reduce the overall network cycle time and therefore, improve the command and control or the force.

a. Collection Enhancements

The Collection node has been improved through advancements in sensor technology [Ref. 1]. Increased range, accuracy, and processing capabilities of sensor systems have significantly enlarged the data flow to the Interpretation node.

b. Interpretation Enhancements

The Interpretation node has been enhanced by data base management and modern video displays. The analyst

can now store, display, and recall data in structured form. This structured form helps the analyst quickly recognize trends, correlate data and, basically, use a systematic approach to the intelligence process. As data flowing into the node is processed, the enemy situation and intentions become visible. The analyst uses this data to present the commander with the best estimate of the situation to use in the Decision node.

With the increasing input to the Interpretation node, and the reduced output flow caused by the inefficiency of the human processor, the need surfaces for some decision aid to correlate and combine data. A system which would help the analyst combine evidence to recognize critical enemy activity, and answer the commander's Essential Elements of Information (EEI) or Priority Intelligence Requirements (PIR) is needed. It is this node of the command and control cycle network which would benefit from the use of an evidence combination support system.

This support system could use some theory of inference, Boolean logic, Bayesian Inference, or Evidential Reasoning to aid the intelligence analyst in combining evidence to recognize the enemy's intent. This evidence combination technique should also allow for the uncertainty that confronts the analyst in his human reasoning process. These specific inference techniques will be discussed in Chapter III.

c. Decision Enhancements

The Decision node is an area which has received substantial attention. In this node the commander and staff plan future courses of action, wargame these options, and then select the course with greatest probability of success. Enemy Courses of Action (ENCOA), Forces Comparison Model (FORCECOM), Contingency Screening Model (CONSCREEN), and

Quick Screening Model (QUICKSCREEN) are current models developed for the commander and staff to aid in this decision process. With the abundance of models available in the military community related to decision making, this node was a good candidate for early enhancement by automation techniques.

d. Action Enhancements

Improving data flow in the command and control system, along with improving battlefield transportation will accelerate the purpose of the Action node by getting troops to the required position on the battlefield in the most efficient manner as possible.

C. THE CRITICAL NODE

Which node in the command and control network holds the key to battlefield success? Clearly, the Collection node will have a very rich environment from which to gather data and transmit to the Interpretation node. Every unit on the battlefield will be forwarding data to be interpreted.

The commander and staff may plan continuously for every conceivable enemy course of action. Forces will be positioned to provide as much deterrent as possible to the enemy.

Sub-units will plan for movement to each battle position for each friendly course of action directed by higher headquarters. Therefore, recognition or interpretation of the enemy course of action becomes the limiting factor in the command and control cycle.

If the initial enemy attack into NATO positions in Western Europe, or any activity during war, can be predicted, the commander can commit his forces, especially the reserve forces kept for this purpose, to repel the

opponent. The Interpretation node is the trigger to this action. It must be accurate, effective, and as streamlined as possible. The entire network process hinges on the critical role of the Interpretation node.

How much of a difference will enhancement of this node by evidential reasoning, or any method for that matter, and the resultant reduction in command and control cycle time, make on the battlefield?

D. BENEFIT OF REDUCED COMMAND AND CONTROL CYCLE TIME, AN EXAMPLE

An example [Ref. 5], is proposed to demonstrate the benefit achieved at a critical battlefield confrontation by reducing the command and control cycle time.

In a European type scenario, units are deployed with the mission of maintaining some force ratio threshold (RTH) in all sectors of operation in order to prevent an enemy breakthrough. Assume there is a predetermined critical sector in which the enemy has chosen to attempt such a breakthrough. The force Ratio in the Critical Sector (RCS) will be computed as:

$$RCS = R(t) \; / \; B(t) \qquad (2.1)$$

where R(t) is the total number of major tank/anti-tank systems in the attacking (Red) force at time (t), and B(t) is the same identifier for the defending (Blue) force. Both force describers are functions of time to allow for attrition and reinforcement by reserve forces. The Red forces are divided between critical and non-critical sectors. The Blue forces have in position a reserve force containing part (percentage of force = x) of the total force. See Figure 2.2 for force dispositions.

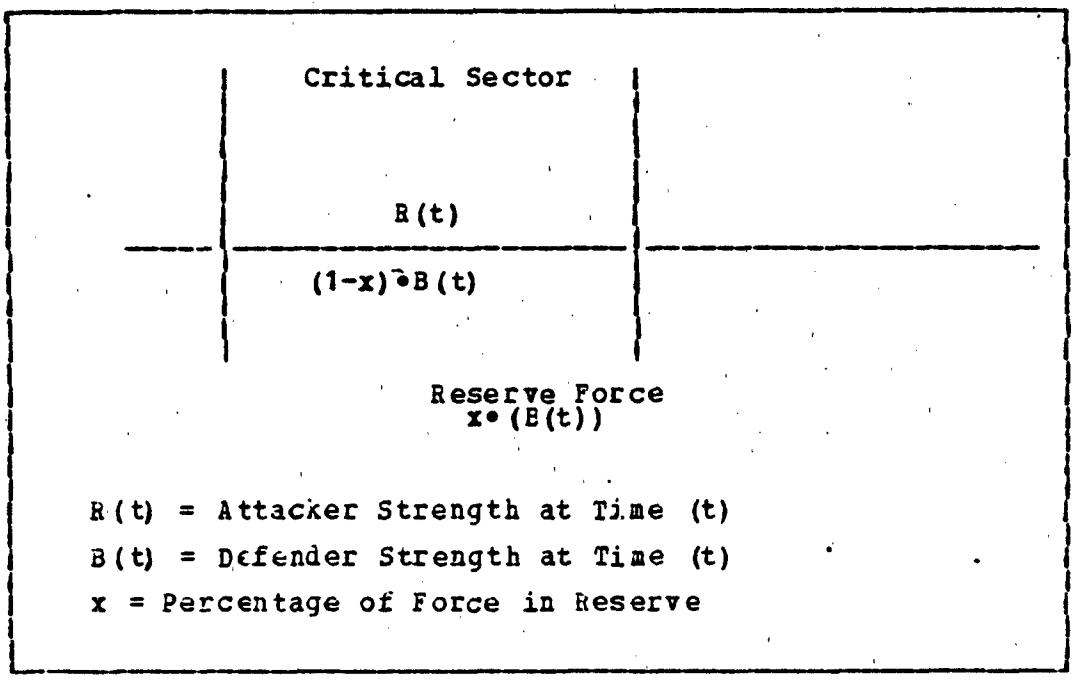

Figure 2.2 Battlefield Layout.

If RTH is the threshold ratio necessary to prevent a breakthrough in the critical sector, the defender must keep RCS ≤ RTH. To accomplish this standoff, the command and control cycle must function in a timely manner so that reserve forces are committed at the correct moment in time and the required threshold ratio is maintained.

Figure 2.3 shows a hypothetical time comparison of the attacker/defender schedules for respective command and control cycles. Figure 2.4 shows hypothetical linear force ratios indicating mission deficiency on the part of the defending force due to the surprise gained by the attacker's commitment in the critical sector.

Reduction of the defender's command and control cycle will increase B(t), increase attacker attrition, and reduce the mission deficiency, see Figure 2.5. The time sequence used in Figures 2.4 and Figure 2.5 correspond to those schedules depicted in Figure 2.3.

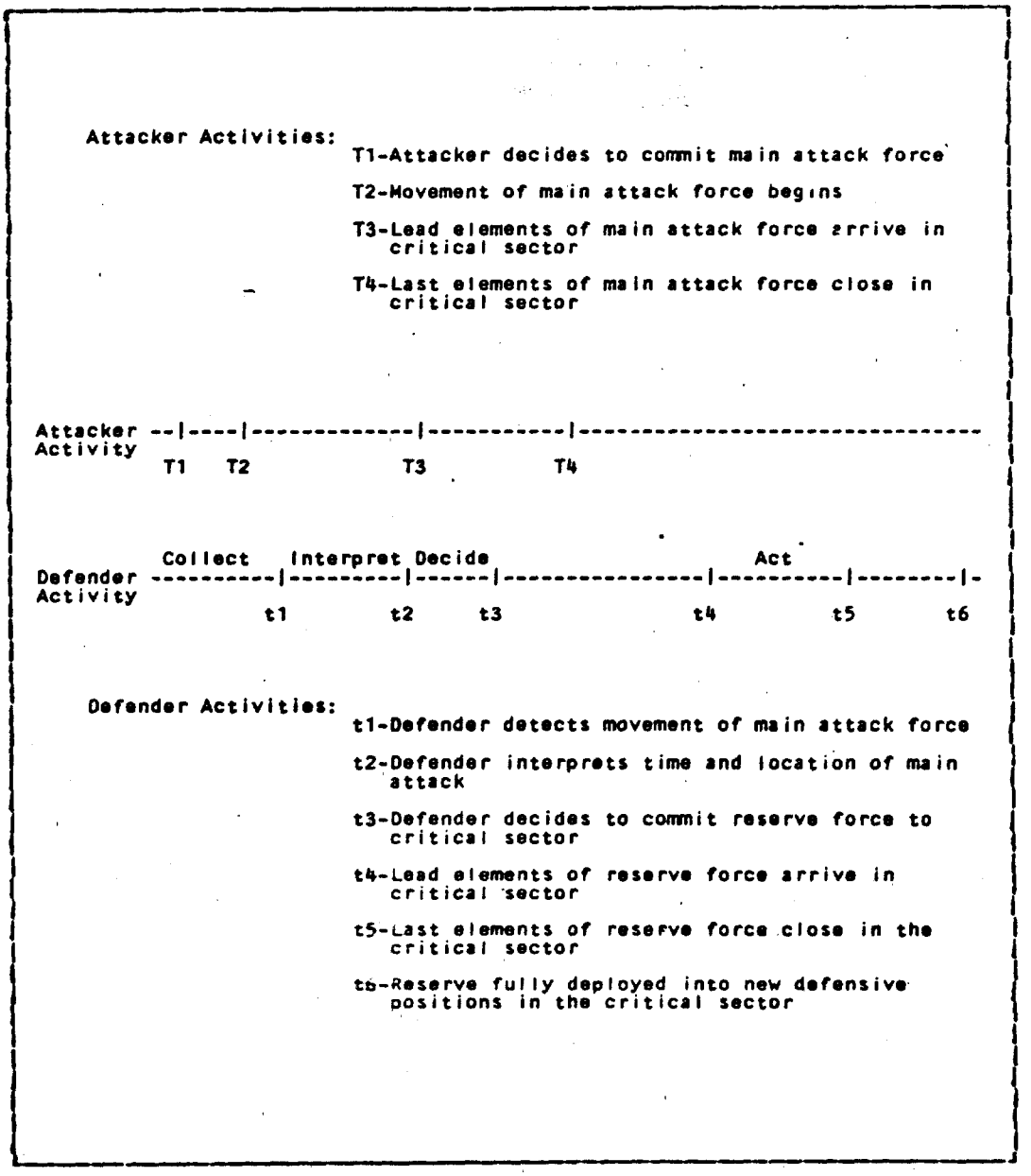

Figure 2.3 Time Comparison of Attacker and Defender Cycles(notional).

Although this is a very simplistic and hypothetical situation, it nonetheless demonstrates the advantages that occur when the command and control cycle is shortened by any

node in the network. The earlier the critical sector can be reinforced, or the deeper the attacker can be interdicted, the more likely the ratio in the sector can be reduced to RTH and a breakthrough prevented. Consequently, any reduction in cycle time can be directly equated to an increase in effectiveness of the force. Therefore, reduction of the Interpretation node's processing time by a decision aid using evidence combination will have a positive effect on the battle.

To further reduce cycle time, it is possible to initially have a contingency plan allowing for a major attack in all sectors of the front. Once the critical sector is identified by the Interpret node, the plan for that sector would be executed. This contingency would almost remove the Decision node from the command and control network. This planning concept would reduce the cycle time by that former amount allocated to the Decision node and further reduce the mission deficiency.

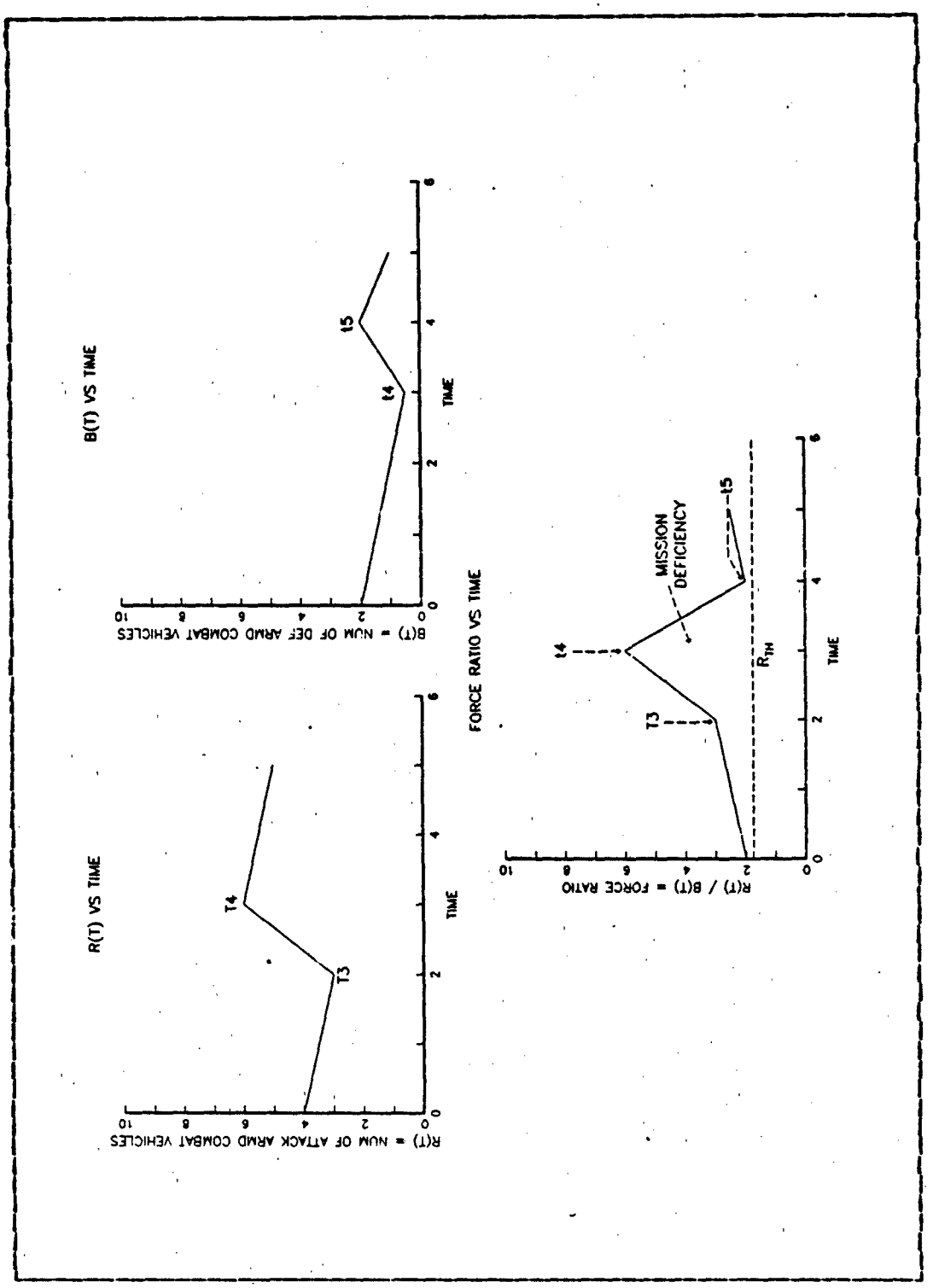

Figure 2.4 Comparison of Attacker and Defender Force Ratios.

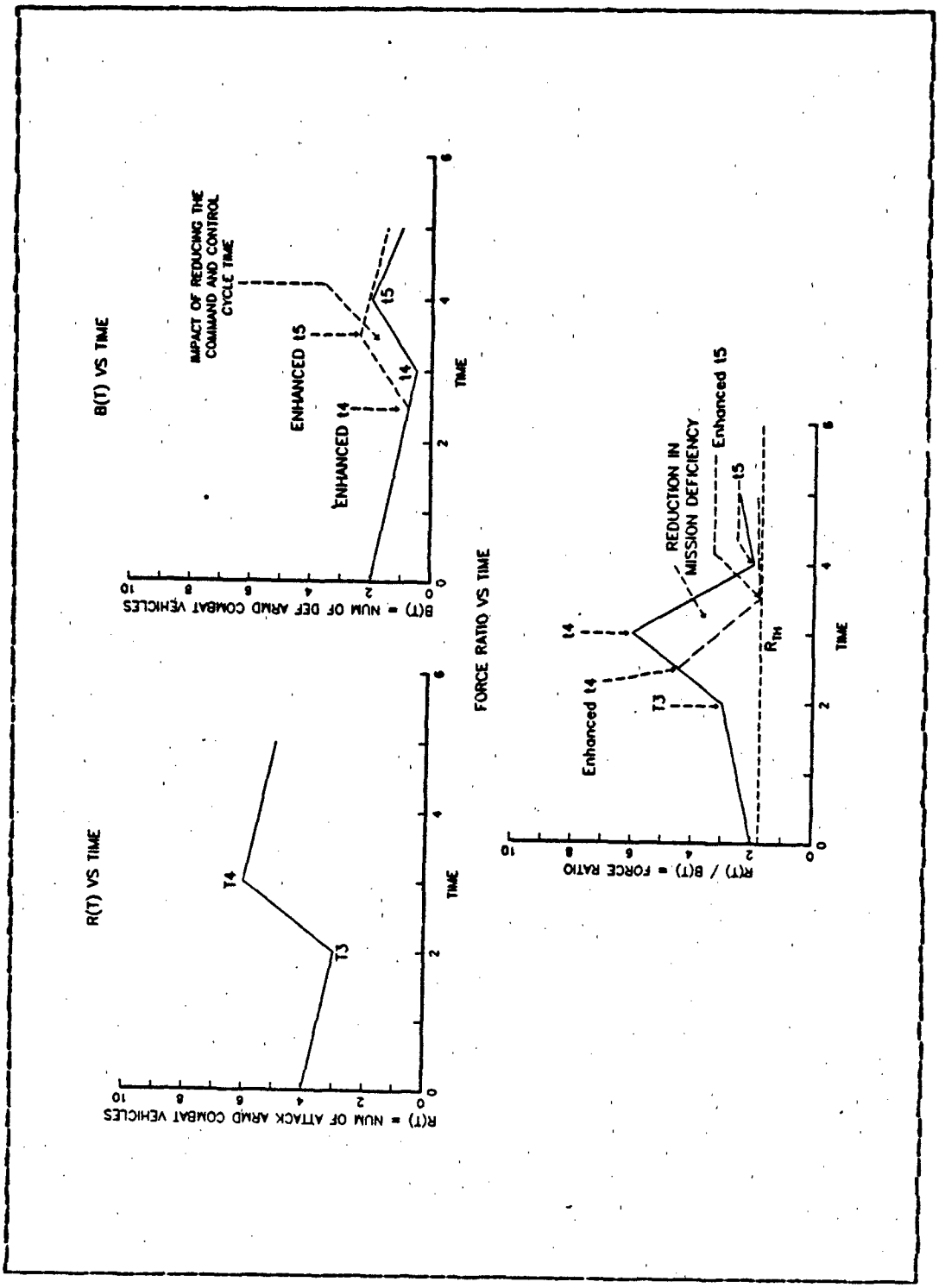

Figure 2.5 Reduction in Defenders Deficiency.

III. EVIDENTIAL REASONING

A. BACKGROUND

As battlefield information flows into the Interpretation node of the Command and Control cycle, it must be combined or fused with prior information to update the understanding of the enemy situation. As previously discussed, this process is currently being accomplished by a man-in-the-loop system that can slow down the entire command and control cycle [Ref. 1]. Methods of evidence combination that can enhance, not replace, the human inference process will now be discussed.

The battlefield situation of Chapter II described the commander's desire to recognize the sector of main attack. The determination of the enemy's intentions can be viewed as a test of hypotheses consisting of all mutually exclusive sector combinations. Given the knowledge that an attack is imminent, such as the first battle in a European scenario, it is the analyst's job to accept or reject these hypotheses. To accomplish this task, the analyst must place values on each likely sector, or hypothesis, indicating the probability of an attack in that sector.

The comparison here of battlefield reasoning to hypothesis testing is logical. The analyst has a set of hypotheses, composed of the sectors of possible attacks, and their multiple conjunctions. These hypotheses indicate attacks over any one, or any combination, of the sectors in the force commander's zone of responsibility.

An example of this battlefield situation will now be described. It will demonstrate the use of an evidential reasoning process in the Interpretation node. This example will be used throughout the chapter.

To begin, suppose the unit using this reasoning enhancement is a U.S. Division. The division's area of responsibility is divided into three brigade sectors (1,2,3). The enemy attack, assuming this is the state of the battle, could occur in Sector 1, 2, or 3. If it occurs near a brigade boundary, or is wide enough to cover more than one sector, the attack could occur in combinations (1,2) or (2,3). If a divided attack occurs, or attacks from adjacent enemy forces occur, then (1,3) is a possibility.

The battlefield, or population, is sampled via the unit's sensors, and the sampled evidence leads the analyst to accept or reject the hypotheses of attack locations. In the case of battlefield sampling, the sampling process takes place over time.

This chapter will discuss the quality of the evidence presented to the decision maker, the use of Boolean and Bayesian methods to evaluate the analyst's hypotheses, and an in depth look at a technique for evidential reasoning based on the Dempster-Shafer theory of evidence.

B. THE QUALITY OF EVIDENCE

In the tactical environment, the Interpretation node will receive information from many sensors, both human and machine. This information is inherently uncertain, incomplete, and sometimes inaccurate. Although less than optimal, this situation is the nature of the battlefield and the nature of the evidence received. The term evidence becomes appropriate here since the information received will be a basis for conclusions or judgements, not a clear answer to any one hypothesis.

1. Uncertain Evidence

Battlefield sensors cannot describe their sample in precise detail and, therefore, create inherent uncertainty in the reports they generate. Some sensors can indicate movement direction, size of the element, or type of unit observed. Other sensors may only report that a unit is moving and may not detect the size or exact location of the target. None can give a complete description of the event. The sensor or operator is always uncertain of many attributes of the target. As Lowrance and Garvey recognize [Ref. 6], the evidence tends to lend varying degrees of support to one or more hypotheses rather than completely specify the event.

2. Incomplete Evidence

The sensor information will also be incomplete. The battlefield sensor can only "view" its assigned sector of search. It can describe what it sees, but cannot lend evidence to what it cannot see on other parts of the battlefield. The analyst should realize that this incompleteness exists, and direct movement of sensors or change sensor search areas to receive a more complete battlefield picture.

3. Incorrect Evidence

The third characteristic of the information collected by the sensor is that it could be incorrect. The operator, interpreter, or soldier reporting could be completely mistaken in their spotting, or the enemy could be using deceptive techniques to confuse the opposing force.

For these three reasons: uncertainty, incompleteness, and incorrectness, the hypotheses or propositions can only be attributed degrees of support based on the evidence received. No one piece of information can be accepted as

complete truth. Therefore, it is the iterative process of combining information from all sources through time, that will lessen the damaging effects of poor-quality evidence and produce good intelligence. To accept or reject the propositions based on a single, cloudy piece of evidence would certainly bias the entire intelligence prediction effort.

What statistical methods can be used to evaluate the evidence in terms of lending support to the acceptance or rejection of the hypotheses? Three methods of evaluation will be discussed. These are: Boolean Logic, Bayesian Inference, and Evidential Reasoning using the Dempster-Shafer theory of evidence combination.

C. TWO COMMON METHODS FOR EVIDENCE EVALUATION

1. Boolean Logic

In Boolean logic the hypotheses or propositions can only be represented as True or False. Varying degrees of support are not accepted, and any information relative to the hypothesis would have to be interpreted as total support or total negation.

As Lowrance and Garvey indicate [Ref. 6], Boolean logic cannot capture the partial belief in hypotheses generated by the coarse evidence received. The battlefield will never be an area for clear cut decisions in black and white, but will always tend towards decisions that deal with the "grayness" of the evidence.

Continuing with the Division example, a report is received of a small unit, an enemy motorized rifle company, moving towards the Forward Line of Troops (FLOT) of Sector 1. This activity could be the advance of the enemy's reserve force indicating a breakthrough attempt, or it could be a feigning action, or only a partial repositioning of

troops. Based on this activity, the analyst cannot, with 100% certainty, predict that the main attack is coming in Sector 1 (True) or that it does not indicate an attack in Sector 1 (False). Nor does the evidence negate or support the possibility of attack in the other sectors.

2. Bayesian Inference

Bayesian methods have been proposed as a basis for several decision aids [Refs. 7,8]. However, these methods have inherent inefficiencies in dealing with disbelief, information supporting the compliment of the hypothesis, and a priori probabilities.

a. Bayesian Formulation

A Bayesian approach [Ref. 7] would consider various hypotheses, such as an attack in Sector 1 (S1) given some datum (D) from a battlefield sensor. The probability that (S1) is true given the data would be:

$$P(S1|D) = P(D|S1) \cdot P(S1) / P(D) \qquad (3.1)$$

where:

$P(S1|D)$ = posterior probability of the hypothesis given the observed datum. (3.2)

$P(D|S1)$ = probability of the datum given the hypothesis. (3.3)

$P(S1)$ = prior probability of the hypothesis before the datum. (3.4)

$P(D)$ = probability of the datum occurring. (3.5)

The task of determining just this last probability of a datum occuring on the battlefield, is more than formidable. Given a fixed set of target describers :

$$TD = (td(1), td(2), \ldots, td(n)) \tag{3.6}$$

Where td(1) and td(2) could be map grid coordinates in easting and northing, td(3) could be the type of unit, td(4) the size of the unit, td(5) the direction of movement, etc. The analyst would then express partial beliefs over TD by distributing belief to the elements of TD.

For example, a report is received about a tank company in the intelligence analyst's area of responsibility. He must now determine the probability of detecting the tank company at the reported location. In this case TD = (td(1),td(2)), the set of grid coordinates in the division zone. Probabilities must be mapped to all grid coordinates in his area which are maneuverable by tanks. The analyst then adds the probabilities for td(1) and td(2) corresponding to the reported grid coordinates of the tank company to determine the probability of the datum occurring.

Probabilisticly, a piece of evidence will map the propositions in TD to the closed interval [0,1]:

$$m : TD \dashrightarrow [0,1] \tag{3.7}$$

where

$$\text{SUM } m(td) = 1, \text{ for td in TD} \tag{3.8}$$

In other words, the conjunction of all evidence from the mapping must equal one, the basis for a probability statement.

Then for any proposition defined over TD, such as the report of the tank company (TANK), the probability of occurrence is:

$$\text{for all td in TD,} \qquad (3.9)$$
$$\text{Prob(TANK)} = \text{SUM } m(td),\ td \text{ in (TANK)}$$

The probability of proposition (TANK) is determined by the sum of the probability of all location possibilities that are elements of the proposition.

It follows that:

$$\text{Prob(TANK)} = 1 - \text{Prob (not TANK)} \qquad (3.10)$$

and since the environment sums to 1:

$$\text{SUM } m(td) = 1,\ td \text{ in TD} \qquad (3.11)$$

Thus all probability not in (TANK) would lie elsewhere in TD, as seen in Equation 3.10. The inherent problem with this approach is that the sensor operator, the intelligence analyst, or an expert, must determine each a priori probability for the partitioning of TD into its elements (td(1), td(2), ... td(n)) given by the mapping m.

This mapping would not be a great problem given a rich data base for a well-defined environment. However, on the battlefield, the sensors will be receiving data on micro events that may only occur once, and depend on time, weather, terrain, or any other target descriptors used in (TD). To compute a priori probabilities given the general nature of the situation would be an endless task and may not be acceptable for time-critical tactical decision making.

Using equation 3.1 and this chapter's example of a main attack against a Division, consider the Bayesian task of determining the probability of an occurrence. If datum (D) is the report of the advancing motorized rifle company, then P(D) is the probability of a such a company advancing. Immediately the analyst is in a predicament. He could make a calculated guess about this value if an extensive data base existed. However, there is no data base, and the analyst definitely does not have the time and may not have the expertise to concern himself with these detailed parameters.

Further, what specific type of company was observed by the sensor? If it is a reconnaissance company, then the probability of this datum occurring could be high. Next, the probability of attack given this datum (P(S1|D)) must be considered. There are now more factors and probabilities with which the analyst must concern himself, most of which are not known.

If the advancing company is a second echelon element, then there would be a low probability of the unit being in this forward area, unless it is the advance of a breakthrough attempt. This probabilistic predicament could go on and on. Because of the many unknowns, Bayesian inference may not be the most desirable method with which to deal with battlefield evidence combination.

Also, a Bayesian supporter would say that if evidence supported two mutually exclusive propositions, and there was no reason to consider either over the other, each should be assigned equal probabilities. So, if evidence supports proposition (X or Y or Z), with probability 0.6, it supports individual propositions (X),(Y),(Z) with probability 0.2. As a result, there is a twofold support of the disjunction of any two of these propositions over the other.

If:

$$P(X \text{ or } Y \text{ or } Z) = .6$$

and

$$P(X) = P(Y) = P(Z) = 0.2$$

then:

$$P(X \text{ or } Y) = 0.2 + 0.2 = 0.4 = 2 \bullet P(Z)$$

However, there was no evidence received to indicate that the disjunct occurrence (X or Y) was greater than the singleton of (Z). The only proposition the evidence supported was (X or Y or Z), and in no way could distinguish it between subsets of that event.

Further problems result from the inability to represent ignorance (lack of support) through Bayesian methods. In the natural reasoning process of the analyst or human sensor lies a critical and distinct difference between lack of support for a hypotheses and support for the compliment of a hypothesis. If (X) and (Y) are the two propositions under consideration, then in a cognitive frame, lack of support for (X) does not necessarily equate to support in (Y).

If:

$$P(X \text{ or } Y \text{ or } Z) = 0.6$$

then in Bayesian terms:

$$1 - P(X \text{ or } Y \text{ or } Z) = P[\text{not } (X \text{ or } Y \text{ or } Z)] = 0.4$$

Of utmost importance, it is critical to recognize that the evidence received was incomplete and this distinction between (X or Y or Z) and not(X or Y or Z) cannot be made. Due the sensor's restricted sampling of the battlefield, the evidence can only support the disjunction, not refute it.

The concerns of representing uncertainty and ignorance while dealing with battlefield-quality evidence

lead to the theory of evidence proposed by Arthur Dempster and Glenn Shafer. This is not to say that Bayesian Inference should not be used on the battlefield. On the contrary, if the probabilities for the use of Bayesian Inference are available and if the analyst can distribute probabilities to single elements, then definitely Bayesian Inference should be used. However, if these conditions are not met, and tremendous assumptions would be required to meet the Bayesian prerequisites, then Dempster-Shafer should be considered as an alternative.

D. THE DEMPSTER-SHAFER THEORY

This theory of evidence combination was conceived by Arthur Dempster [Ref. 9], and later developed by one of Dempster's students, Glenn Shafer [Ref. 10]. It is a theory of evidence because it deals with support of propositions based on evidence. It allows for quantifying ignorance, or lack of knowledge, as well as uncertainty. It uses a term, plausibility, to indicate lack of belief in a proposition rather than suggesting support of the compliment of the proposition. The term, Belief, is used to indicate support for any proposition.

1. Formulation

The Frame of Discernment, the set of all mutually exclusive propositions, is represented by: THETA. The domain of THETA is the set of all possible subsets of THETA.

$$\text{Domain Size} = 2 \exp |\text{THETA}| \text{ subsets} \qquad (3.12)$$

(2 raised to the magnitude of THETA subsets)

An example of a Frame of Discernment would be the three brigade sectors of the division zone in which the main enemy attack could occur. In this case THETA = (1,2,3), assuming 3 brigades in a division.

The domain would be :

(1,2,3) (1,2) (1,3) (2,3) (1) , (2), (3) , ()

which are all the subsets of THETA. Although the Null Set, (), will always be a subset of THETA, it will never have Belief and will not be listed as a subset in the rest of the calculations in the chapter.

The mapping of probability assignments to the propositions is done by a Basic Probability Assignment (BPA), referred to as Mass (M), which satisfies:

M(Null Set) = 0 (3.13)

and

SUM M(i) = 1, i in THETA (3.14)
(the sum of Masses over THETA = 1).

Support for any proposition (X) is given by Belief(X), or Bel(X), defined as:

Bel(X) = SUM M(i), i in X (3.15)

Belief is the sum of all the Masses of all subsets of the proposition. The Belief of (X) is the measure of the Mass constrained to stay somewhere in (X) [Ref. 12]. Belief of an attack in sectors (1 or 2) would equal:

Bel(1,2) = M(1,2) + M(1) + M(2)

Related to Belief is Plausibility or the Upper Probability Function defined as the total probability Mass that has potential for moving into (X).

$$Plaus(X) = 1 - Bel(\text{not } X) \tag{3.16}$$

In the example,

$$Plaus(1,2) = 1-Bel[\text{not}(1,2)] = 1-Bel(3)$$

The Doubt of (X) or Dou(X):

$$Dou(X) = Bel(\text{not } X) \tag{3.17}$$

is the measure of probability forced to stay out of A.

$$Dou(1,2) = Bel[\text{not}(1,2)] = Bel(3)$$

It follows that:

$$Plaus(A) \geq Bel(A) \geq M(A) \tag{3.18}$$

In contrast with Bayesian Inference, Dempster-Shafer allows motion of Masses throughout the frame or discernment since each Mass need not be constrained to single elements within THETA. Therefore, no requirement exists to commit Masses to elements past the level of recognition contained in the report, constrained by the sensor's limited capabilities.

If evidence received indicates an attack in Sector 1 or Sector 2, the support need not be divided between the two propositions, (1) and (2). If movement of enemy forces towards the front occurs on a road bisecting the two zones, it is not necessary to say that:

$$P(1) = P(2) = 1/2 \bullet P(1,2)$$

but instead, the evidence can be assigned to the superset: (1,2).

By using the two values of Belief and Plausibility, support for a proposition or hypothesis can be expressed by an interval as follows.

$$\text{Evidential Interval (EI)} = [\text{Bel}(X), \text{Plaus}(X)] \qquad (3.19)$$

where the difference, Plaus(X) - Bel(X), can be referred to as the Ignorance remaining about (X):

$$\text{Ig}(X) = \text{Bel}(X) - \text{Plaus}(X) \qquad (3.20)$$

If Ig(X) = 0, there exists no Mass available to move into (X). Further, if Ig(X) = 0 for all propositions, the system is Bayesian. This is true since this would require that all Masses be distributed to singletons in the frame of THETA. For this reason, Bayesian Inference can be described as a subclass of the theory of Belief functions [Ref. 14]. Table 2 show some examples of these Evidential Interval values.

TABLE 2

EXAMPLES OF DEMPSTER-SHAFER EVIDENCE INTERVALS

X(0,1)	=>	No knowledge at all about (X).
X(0,0)	=>	(X) is false.
X(1,1)	=>	(X) is true.
X(0.25,1)	=>	Evidence provides partial support for (X).
X(0,0.85)	=>	Evidence provides partial support for (not X).
X(0.25,0.85)	=>	Probability of (X) is between 0.25 and 0.85;i.e., the evidence simultaneously provides support for both (X) and (not X).

In the continuing Division example, a report is received indicating a strong possibility of attack in Sector (1 or 2) represented as (1,2) and a slight possibility of attack in Sector 3, represented as (3). The analyst assigns Mass(1,2) = 0.6 and Mass(3) = 0.2. The remainder of Mass (0.2) cannot be assigned elsewhere, so is assigned to the set representing the entire frame, Mass(1,2,3) = 0.2. These Mass values must be determined in some way by the analyst. He may use probabilities derived from prior analysis of enemy tactics if available, or may use his own judgement based on knowledge of enemy tactics. The latter method will be further discussed in Chapter V. For now assume that the analyst has determined these values.

Belief is now limited to those sets whose subsets have Mass or:

Bel(1,2,3) = M(1,2,3) + M(1,2) + M(1,3) + M(2,3)
+ M(1) + M(2) + M(3)
= 0.2 + 0.6 + 0.0 + 0.0 + 0.0 + 0.0 + 0.2 = 1.0

Bel(1,2) = M(1,2) + M(1) + M(2) = 0.6+ 0.0 + 0.0 = 0.6

Bel(2,3) = M(2,3) + M(2) + M(3) = 0 + 0 + 0.2 = 0.2

Bel(1,3) = M(1,3) + M(1) + M(3) = 0.0 + 0.0 + 0.2 = 0.2

Bel(3) = M(3) = 0.2

Bel(1) = Bel(2) = 0.0 (They have no Mass assigned)

As seen in the above example, since THETA exhausts all proposition possibilities, the Belief in THETA is always equal to 1.0. The Plausibility and Evidential Interval for (1,2) would be:

Plaus(1,2) = 1 - Bel(not(1,2)) = 1 - Bel(3)
= 1 - 0.2 = 0.8

Evidential Interval = [Bel(1,2),Plaus(1,2)]
= [0.6,0.8]

The Ignorance would be:

$$Ig(1,2) = 0.8 - 0.6 = 0.2$$

The Evidential Intervals (EI) for all the sets are:

$$EI(1,2) = [0.6, 0.8]$$
$$EI(2,3) = [0.2, 1.0]$$
$$EI(1,3) = [0.2, 1.0]$$
$$EI(1) = [0.0, 0.8]$$
$$EI(2) = [0.0, 0.8]$$
$$EI(3) = [0.2, 0.4]$$

2. Combination of Evidence

Dempster-Shafer allows for the combination of evidence from knowledge sources or sensors on the battlefield. Given two Mass assignments M1 and M2 new Masses are computed by the orthogonal sum M1 @ M2 (where @ represents orthogonal sum) defined as:

$$M(\text{Null Set}) = 0.0 \qquad (3.21)$$

$$M(A) = \{SUM [M1(X) \bullet M2(Y)]\}/K \qquad (3.22)$$

where $(X \,\&\, Y) = (A)$; ($\&$ represents intersection)

$$K = 1 - SUM [M1(X) \bullet M2(Y)], \quad (X \,\&\, Y) = \text{Null Set}$$
$$= SUM [M1(X) \bullet M2(Y)], \quad (X \,\&\, Y) \neq \text{Null Set}$$

From Equation 3.21, the Mass assigned to the null set must equal 0. This is accomplished by the normalization of the Masses assigned to all other sets. The normalization factor, K, is equal to 1 - the Mass assigned to the null set after combination. K is also equal to the sum of the Masses assigned to the subsets of THETA (less the Null Set) after combination.

'The new Mass, or combined Mass assigned to (A), is the orthogonal sum of the Masses (divided by the normalization factor) where (X & Y) = (A). This orthogonal sum combination technique can be further shown by the unit square in Figure 3.1.

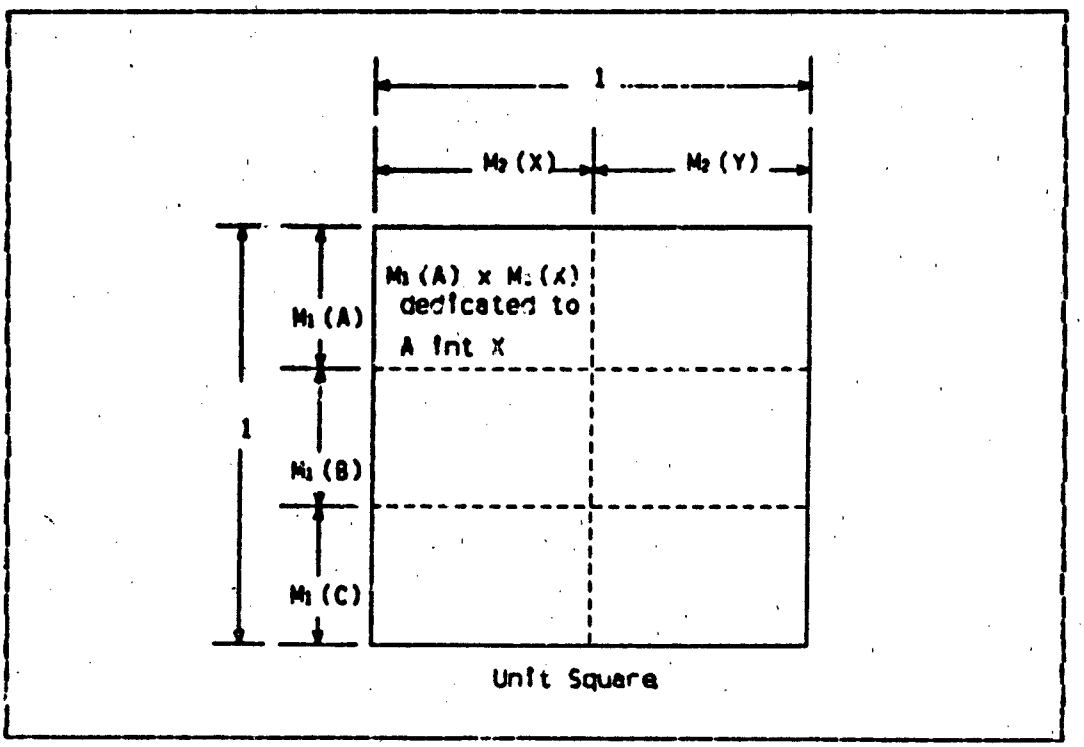

Figure 3.1 Unit Square Example of Orthogonal Sum.

Continuing with the example, a second report is received by the division intelligence analyst for which he assigns Masses as follows:

M2(1,3) = 0.3
M2(2) = 0.3
M2(1,2,3) = 0.4
(where M2 designates second report).

Figure 3.2 shows the orthogonal sum of these example Masses.

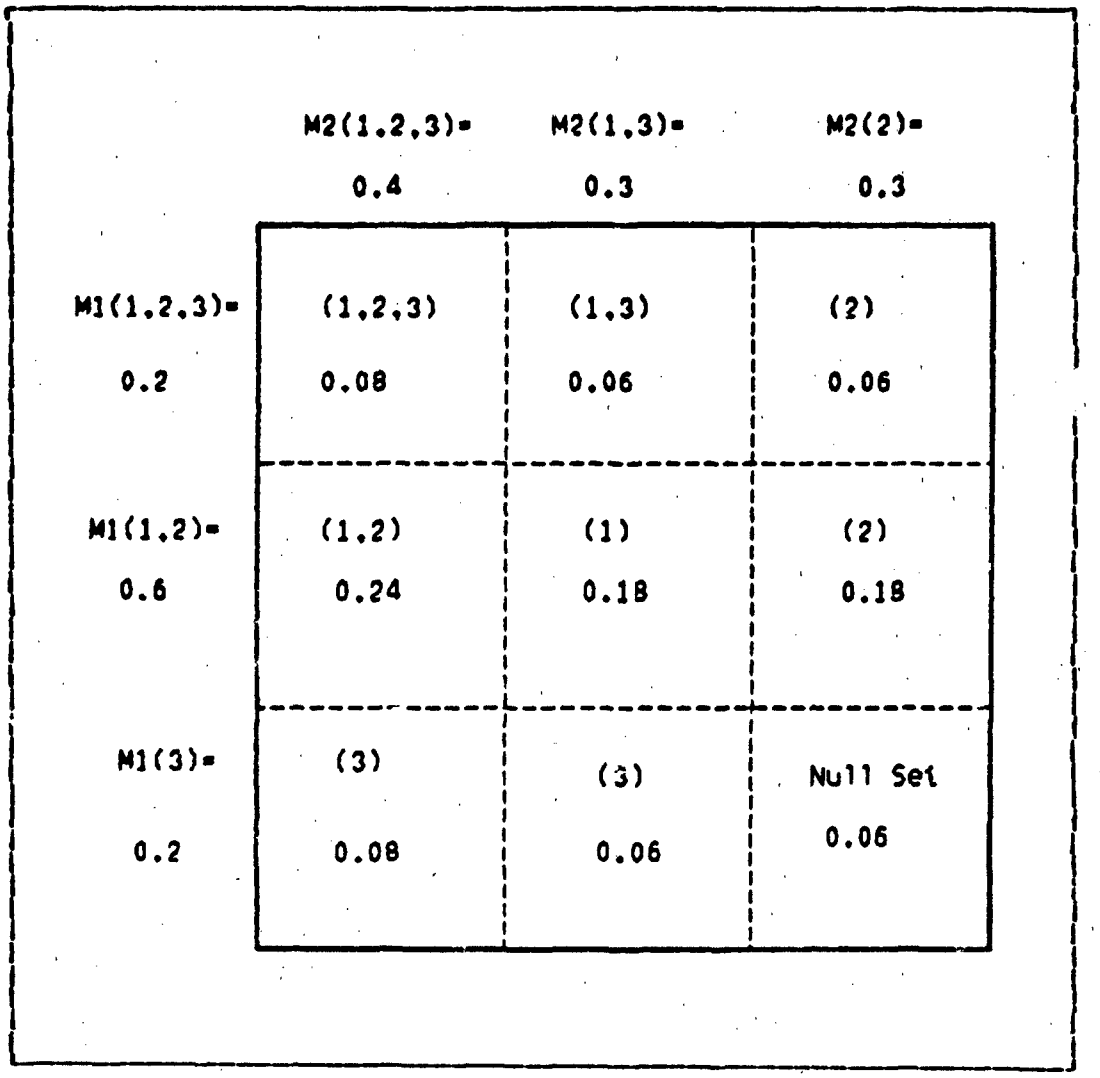

Figure 3.2 Orthogonal Sum of Example Masses.

The Normalizing Factor (K) in this example is:

$K = (1-0.06) = .94$

(1-Mass assigned to the Null Set)

New normalized Masses are:

$M(1,2,3) = .08/.94 = .085$

$M(1,2) = .24/.94 = .255$

$M(1,3) = .06/.94 = .064$

M(2,3) = 0.0 (not an intersection set
in orthogonal sum)
M(1) = .18/.94 = .192
M(2) = (.06 + .18)/.94 = .255
M(3) = (.08 + .06)/.94 = .149

New Beliefs are computed as follows:

Bel(1,2,3) = .085 + .255 + .064 + .192 + .255 + .149
= 1.0
Bel(1,2) = .255 + .192 + .255 =.502
Bel(1,3) = .064 + .192 + .149 = .405
Bel(2,3) = 0.0 + .255 + .149 = .404 (has belief even
though no Mass,due to Mass in its subsets)
Bel(1) = .192
Bel(2) = .255
Bel(3) = .149

The new Evidential Intervals are:

EI(1,2) = [.502,.851]
EI(2,3) = [.404,.808]
EI(1,3) = [.405,.745]
EI(1) = [.192,.596]
EI(2) = [.255,.595]
EI(3) = [.149,.498]

These results show strongest Belief in the pair (1,2) due mainly to the assignment of Mass = .6 to this set from M1. Belief in (1,3) and (2,3) are approximately equal, .405 and .404 respectively. Belief for single elements is strongest for (2) due mainly to its Mass assignment from M1. The final Beliefs follow from the Mass assignments, which is appealing.

Also note the changes in the Evidential Intervals after combination. The intervals have narrowed for most

sets and no set has a Plausibility of 1.0. The Ignorance (Plausibility - Belief) for all sets except (1,2) has decreased.

3. Independence of Knowledge Sources

A point of interest in Dempster's rule of combination is the independence of knowledge sources or independence of reports from the same source. How does this method deal with multiple sightings from same or different sensors?

If the sighting or sensor data come from different knowledge sources, then this evidence can be considered collected from independent elements. Sensors will occupy different terrain positions and have varying operating characteristics and capabilities. As such, these knowledge sources will derive enough independence for this method to produce desired results.

If the same sensor reports the same data, these reports are not independent in the sense that they come from the same source. However, if the same sensor reports on the same unit of activity, but the location or any other reportable characteristic of the unit changes, then this is considered sufficiently independent. An example would be reports on movement direction of a target. This reporting would be considered independent for the purpose of indicating a confidence towards or away form a hypothesis [Ref. 11].

If the same sensor is reporting on an activity that does not move, a nigher headquarters location, then this should be used as new evidence. It confirms headquarter location and also the fact they are not moving (shown by many reports from the same sensor) and therefore may indicate something to the analyst to be included in the evidence combination.

4. Some Points of Contention with Dempster-Shafer

Conflict over the Dempster-Shafer theory of evidence arises from the following:

a) Normalization. This is the normalizing of resultant Masses after combination. Mass that would go to the null set is ignored and Masses going to THETA are normalized.

b) Total Conflict. When total conflict of evidence occurs, Mass for all sets sums to zero. The normalizing factor (K) is equal to zero, and the attempt to divide by K, of course, fails. This case is the same as sending all Mass into the null set.

However, there are several explanations to reduce concern over these events.

(1) Normalization. Consider Masses tending to support conflicting propositions, propositions that have no intersection. The Mass in this event flows into the null set indicating conflict between Masses from knowledge sources. Since this conflict occurs due to uncertainty about the situation at hand, as more evidence is received conflict will diminish.

Also, as certainty toward the correct proposition increases so does ability to decrease the number of elements in THETA. Then, less conflict will occur in the evidence from sensors and the need for an evidence combination technique that deals with uncertainty diminishes. But, this is not the case on the battlefield, where the sensors are spread over great distances across division fronts. These sensors are directed toward different areas of the front and a great diversion of information is desired by the intelligence analyst through a wide variety of contacts.

Normalization, then, is just a means of dealing with the conflicting nature of evidence. The

measure or weight of conflict is represented by the magnitude of the normalizing factor (K). K indicates how much conflict occurs between the current Mass assignment and the resultant Mass values of previous assignments. If the set of all possibilities is the Frame of Discernment, the null set cannot occur. The Mass assigned to it represents conflict, a normal occurrence when so many sensors report so varied data. The Mass cannot remain in the null set, therefore all Masses are normalized.

A large measure of conflict occurs in the following example:

M1(A) = .99
M1(B) = .01
M2(C) = .99
M2(B) = .01

M1 @ M2 yield:
M1(B) = .0001
M1(Null Set) = .9999

Normalized results are:
K = 1 - .9999 = .0001
M1 @ M2 (B) = .0001/.0001 = 1.0, a questionable result

However, if in fact (A,B,C) were the only possible results, this conclusion is logical. Only through combining this battlefield-quality evidence and dealing with inherent uncertainty can the analyst reach conclusions about the hypotheses. If the rigorous methods like Bayesian Inference cannot be used, then Dempster-Shafer seems to offer a logical alternative.

(2) Total Conflict. The second shortcoming, total conflict of evidence, is caused by total combined Mass going to the null set aborting any combination effort. This

occurs by assigning total Mass from two knowledge sources to sets with no intersection, such as:

M1(A B) = 1.0
M2(C,D) = 1.0
M1 @ M2 ==> K = 0

since:

[(A,B) & (C,D)] = Null Set

However, is it possible to be 100% certain of (A,B), yet at the same time, to be 100% certain of (C,D)? No, this situation and resulting conflict are unacceptable and can only be resolved by the realization that,again, the analyst is dealing with uncertainty. Inaccuracies in knowledge sources cannot allow assignment of probability Masses in this manner.

As further proof, consider the following example [Ref. 12]. A fair die is rolled and knowledge source one places all Mass in the proposition that the number is even, M1(EVEN) = 1.0. Conflicting evidence from knowledge source two places all Mass in the proposition that the number is odd, M2(ODD) = 1.0. The result of combination assigns all Mass to the null set since (EVEN & ODD) = Null Set, and the combination fails.

This example shows that this occurrence would tend to violate the assumption of uncertainty of the evidence and also falsify any logical reasoning process. The analyst cannot let this occur under any method of evidence combination.

E. BAYES' RULE OF CONDITIONING

In a battlefield situation, using Bayes' conditioning rule to combine evidence does not seem to generate more

satisfying results than those achieved by the Dempster-Shafer method.

A probability distribution m would be transformed by Bayes' Rule to m' by the receipt of additional information. It is necessary to restrict the domain of m to elements of (X) when using this rule. Bayes' rule is:

for all i in THETA: (3.23)

$m'(i) = 0.0$, if i not in (X)

$m'(i) = m(i)/(1-k)$, if i in (X)

where:

$k = \text{SUM } m(i) < 1$, i in (X) (3.24)

Or from equation 3.1:

$$P(S1|D) = P(D|S1) \cdot P(S1)/P(D)$$

Using the results of the prior knowledge source report outcome, P(S1), P(S1|D) is updated by the probability of the new datum, P(D), and probability of the datum given the hypothesis, P(D|S1).

A more tractable form of the equation can be constructed using likelihcod ratios where:

$$P(S1|D)/P(S2|D) = [P(S1)/P(S2)] \cdot P(D|S1)/P(D|S2) \text{ or}$$

$$P(S1|D)/P(S2|D) = [P(S1)/P(S2)] \cdot L(D|S1:S2)$$

where $L(D|S1:S2) = P(D|S1)/P(D|S2)$ and is called the likelihood ratio favoring hypothesis S1 over S2.

Now since P(S1|D) + P(S2|D) must equal one, the final values for posterior probabilities are determined by their ratio. It is no longer necessary to determine the probability of the datum (P(D)), nor is it necessary to assess the probabilities P(D|S1) and P(D|S2) if the likelihood ratio L(D|S1:S2) is used. However, it may be easier to

compute the likelihood ratio based on the probability P(D|S), which will be demonstrated in an example.

The following example using this Bayesian method is presented. Three hypotheses under consideration are:

S1 = attack in Sector 1
S2 = attack in Sector 2
S3 = attack in Sector 3

The first difficulty encountered using this approach is determining prior probabilities. Should they be calculated using the first piece of evidence, or be based on other knowledge. In this example there is no reason to favor one over the other, so assign equal probabilities: P(S1) = P(S2) = P(S3) = 0.333.

The first piece of evidence received (D1) indicates strong possibility of attack in Sector 3 (S3). To ease computation, all likelihood ratios will now be based on Sector 3, i.e.: L(D1|S1:S3), L(D1|S2:S3) and L(D1|S3:S3) will be used. Table 3 column L(D1|S(i)|:S3) lists the likelihood ratios based on this first piece of evidence.

The likelihood ratios can be computed in the following manner. Datum 1 (D1) indicated a high probability of attack in Sector 3, say P(D1|S3) = 0.8. To compute likelihood ratios, a comparison must be made with the other hypotheses. If D1 indicates a small probability of attack in Sector 1, (S1), say 0.10, then the likelihood ratio L(D1|S1:S3) = .1/.8 = .125. As previously stated, this likelihood ratio could have also been determined by saying that the probability of attack in Sector 3 based on the datum is eight times greater than attack in Sector 1.

The ratio for Sector 2 and 3 will be .2/.8 = .250, or attack in Sector 3 is four times more likely than attack in Sector 2, based on the datum.

The next piece of evidence received (D2) assigns the following likelihood ratios:

$$L(D2|S1:S3) = .70/.50 = 1.4$$
$$L(D2|S2:S3) = .80/.40 = 1.6$$
$$L(D2|S3:S3) = 1.0$$

Now update all posterior probabilities for comparison using the three equations and three unknowns:

$$P(S(i)|D)/P(S3|D) = [P(S(i))/P(S3)] \cdot L(D1|S(i)):S3) \cdot L(D2|S(i):S3), \text{ for all } i$$

and solve simultaneously.

Column P(S(i))|D) of Table 3 (where D represents all evidence received) shows the final posterior probabilities. Now comparisons of hypotheses may be made using these posterior probabilities.

TABLE 3

BAYESIAN INFERENCE EXAMPLE USING LIKELIHOOD RATIOS

	P(S(i))	L(D1|S(i):S3)	L(D2|S(i):S3)	P(S(i)|D)
S1	.333	.125	1.4	.111
S2	.333	.250	1.6	.254
S3	.333	1.0	1.0	.635

P(S(i)) = Prior probability
L(D1|S(i):S3) = Likelihood Ratio from first datum
L(D2|S(i):S3) = Likelihood Ratio from second datum
P(S(i)|D) = Posterior probability

There are several problems encountered using this approach. First, how should the prior probabilities be chosen? If they are equally likely, then equal probabilities could be assigned as done in this example. It may be better to use the first piece of evidence receive, but these priors must sum to one. If the initial evidence indicates a strong possibility of attack in Sector 1, but lends no support to the other hypotheses, how should prior probabilities be assigned to the other hypotheses? Some prior probability must be assigned to the other hypotheses, if not, they will always have a posterior probability of zero.

Second, there can be a lack of consistency in assignment of likelihoods. There is no constraint to the magnitude of the ratio. In the example, a likelihood ratio of 1.4 was used. Yet, if a strong indicator of attack (.9) was compared to a hypothesis with a very slight chance of attack (.001), then a likelihood ratio of .9/.001 = 900 would occur. This lack of constraint on the magnitude of the likelihood ratio may lead to inconsistencies as the inference progresses through many likelihood ratios.

Shafer and Tversky [Ref. 13] remark that traditional Bayesian theory has been concerned with what they call observation design. This design deals with outcomes of statistical experiments. In the experimental space, the analyst knows the possible outcomes and answers. Prior probabilities for parameters can be assessed in advance. Bayesians have gradually extended their experimental space to the area of data analysis where probabilities are not so clearly defined. This possible over extension of Bayesian Inference could lead to its partial demise in the battlefield hypothesis space.

This section has not been presented as a critique of the Bayesian method, but as an insight that there are problems with Bayesian methods as with Dempster-Shafer.

IV. A COMPUTATIONAL VIEW OF DEMPSTER-SHAFER

A major drawback to the use of Dempster-Shafer has been the long calculation time required due to its computational complexity. For example, the computation of Belief requires time exponential in |THETA| [Ref. 12]. As part of this thesis research a decision aid was created to assist in the Dempster-Shafer computations. Appendix A contains a description of the aid, as well as, a listing of the PASCAL code. This aid should not be considered a fully operational military decision aid, but rather a prototype or example of an automated evidence combination technique.

The aid is designed to lead the user through the steps necessary to set up and use the Dempster-Shafer theory. The aid's output has been verified by comparison with manually-computed solutions of problems using Dempster-Shafer. Identical results were achieved. The output of the model used for these checks was Belief, Plausibility and Mass.

Although this exponential computational time factor of Dempster-Shafer has been discussed in length, [Refs. 6,12,15], no actual computational data was found to support it. An additional benefit of the aid was the ability to now record these computational times. Also discovered through the use of the aid, was the memory limitation of the computer after all subsets of the Frame of Discernment were enumerated.

Reduction in the computational complexity of Dempster-Shafer will be addressed in this chapter, but first, a brief discussion of the Dempster-Shafer aid created will be presented. Efficiency of the aid in terms of computational time and memory requirements will be described in more depth. Then methods of reducing computational

complexity presented by Barnett [Ref. 12], and Gordon and Shortliffe, [Ref. 15], will be discussed. A third method of reducing computational time, Multiple Frames of Discernment, will be presented.

A. A DEMPSTER-SHAFER DECISION AID

1. Description Of The Aid

The aid is designed for a user familiar with the Dempster-Shafer theory. The user will first see a screen informing him of the theory used in the aid. Next, the size of the Frame of Discernment is requested. A letter of the alphabet is assigned to each member of the Frame of Discernment. If a Frame had five members, THETA would be represented by (A,B,C,D,E), or 7 members THETA = (A,B,C,D,E,F,G) etc. The user is then told the Frame will be represented by these letters.

All subsets of the Frame are generated by procedure "Generate". For each subset, this procedure creates a PASCAL record structure that contains the items of Belief, Mass, and Plausibility may be stored. The sets are sorted by size by procedure "Quicksort" to assist in the search efficiency throughout the program. Next, the aid informs the user of the item number for data input. The item number is just a means of keeping track of the number of loops through the program, which equals the number of data items combined.

The user then enters the set for which he wants to assign Mass. Masses are assigned for all desired sets and then combined by procedure "Combine" using the Dempster-Shafer theory.

The user is then asked if he desires Beliefs to be computed. If so, Belief and Plausibility are computed by procedure "Belief" and displayed. The program then returns to the input mode and will cycle until ended by the user.

2. Efficiency Of The Aid

As previously stated, a drawback to the use of Dempster-Shafer is the time required to compute Belief. As discussed in Chapter III, Belief for a set requires the summing of the Masses of all subsets of the parent set. Figure 4.1 shows execution time for the computation of Belief by the aid, as well as, times for the other major program procedures.

The execution time for procedure Generate increases exponentially. The procedure recursively generates subsets and must check for repetition of sets as it proceeds. The number of subsets is exponential in THETA, 2 exp |THETA|. As the size of THETA grows, the number of subsets grows exponentially, and also, the number of repeat subsets to be checked and eliminated increases. Fortunately, Generate is only executed once at the beginning of the session. It is possible to eliminate the generation of subsets altogether by reading in subsets from hard disk or floppy. All subsets for various sizes of Frames of Discernments could be stored and simply read at the beginning of the program.

Unfortunately, the execution time problem for Belief computations is not so easily solved. Belief will be computed whenever the user desires a status of all the subsets. As the number of subsets grows, this computation may cause the user an unacceptable waiting time to view Belief. During this time no input can be made for new Masses. The way this program is designed, multiple data entries can be made without involving the time intensive Belief procedure. In a battlefield scenario, the Belief computation would be done only when conclusions were required, not after each data input.

Perhaps a greater limitation in using micro computers to solve many real world problems is the large

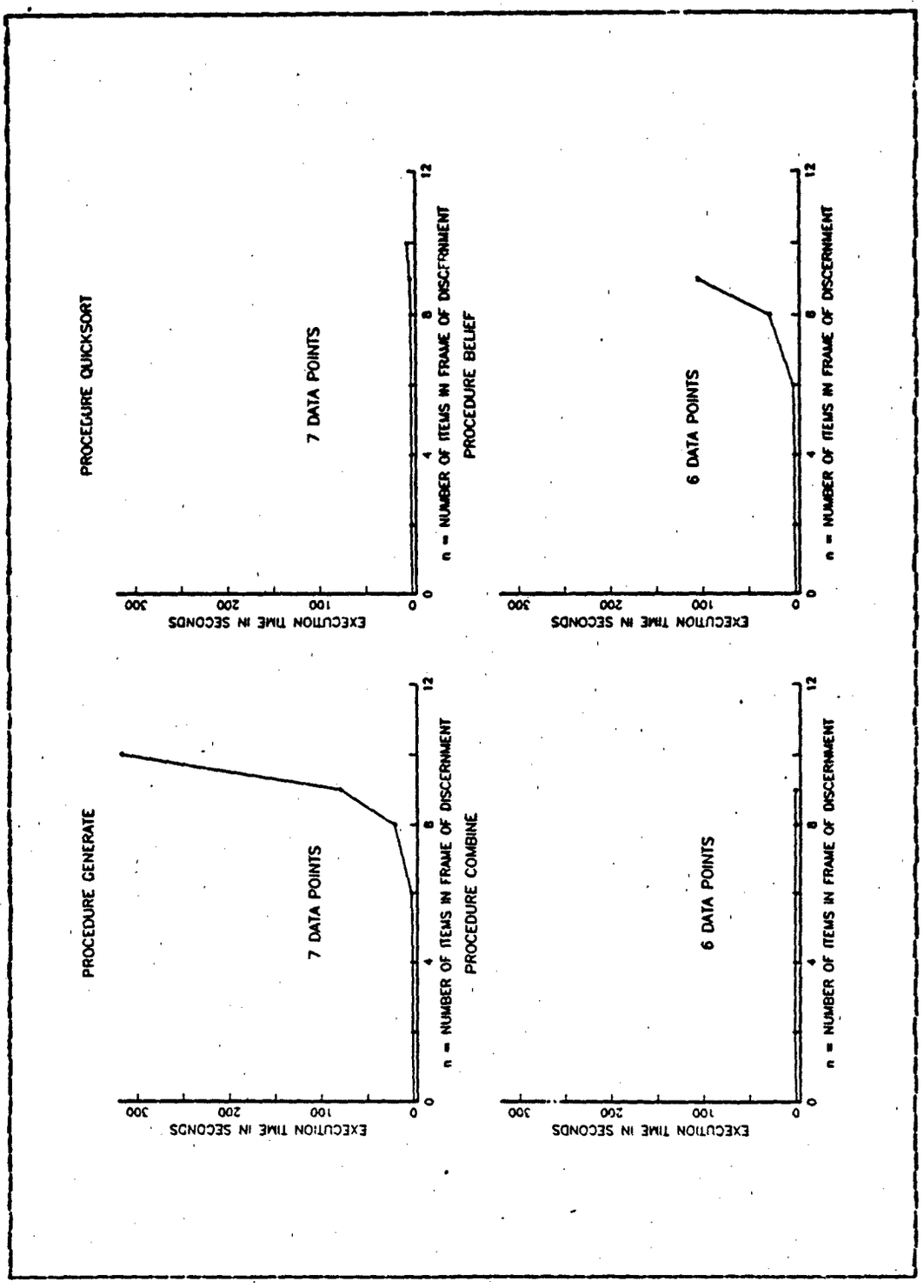

Figure 4.1 Execution Times Of The Aid.

memory requirements demanded by Dempster-Shafer. With this particular coding scheme using a PASCAL record to represent each subset, memory limitations (320K RAM) were reached at a Frame of Discernment size of 9, which equals 2 exp 9 = 512 subset of THETA.

The military examples used so far in this thesis have had a Frame size of three elements and total size of 2 exp 3 = 8. As seen in Figure 4.1, no noticable time delay occurs at this level, which is also well within memory storage limitations. However, the selection of Frame sizes above this level might cause unacceptable delays, especially in real-time battlefield applications.

The three methods for reduction in complexity mentioned in the introduction will now be discused.

B. BARNETT'S METHOD, SINGLETON HYPOTHESES AND THEIR COMPLEMENTS

Barnett, [Ref. 12], showed that if all the subsets of the Frame of Discernment can be reduced to singleton hypotheses and their negations, computational time will be reduced from exponential to polynomial order.

Before proceeding, a new military example will be used to demonstrate the formulation of Barnett's method. The old example of sector of enemy attack is not well suited for this method since it is difficult for the analyst to reduce the scope of the problem to singleton hypotheses and their complements.

A suitable military example for the Barnett method is the analysis of friendly axes of attack or enemy avenues of approach. Assume that the enemy or friendly force must choose the best approach into the combat area, given four avenues A, B, C and D. These singleton hypotheses and their complements will be the only sets considered. This example

will prchibit the analyst from assigning Mass to any combination of elements, such as (A,D), though he may assign Mass individually to (A) and (D). Mass may be assigned in a sense to (B,C,D) by assigning Mass tc (not A).

If the problem at hand can be reduced to subsets using only the hypotheses and their complements, this method does not restrict the analyst. He will seek reports about the avenues of approach into his sector or responsibility. These reports will consist of information about the terrain, weather conditions, natural and man-made obstacles, etc. The analyst will then assign Mass to each hypotheses based on the information and infer the most likely enemy approach or best friendly axis.

1. Formulation Of Barnett's Method

Three steps will be used to represent the formulation of Barnett's method:

a) Combinatien of elements and complements.
   i) Combine evidence confirming each singleton hypothesis.
   ii) Combine evidence disconfirming each singleton hypothesis.
   Step 1 results in the formation of 2•N Belief functions (N=|THETA|), one for each element and one for each element complement.
b) Combination or element pairs. Combine the confirming and disconfirming evidence for each element. This step forms N Belief functions, one for each element of THETA.
c) Combination within THETA. Combine all elements of THETA tc produce one Belief function.

Each step of this method will now be discussed. Only three avenues of approach (A,B,C) will now be considered to simplify the computations.

a. Step 1: Combination Of Elements And Complements

For each singleton, and for each complement, the Mass assigned by all evidence received is combined. Using avenue of approach (A) from the example, the evidence confirming (A) after receipt of two pieces of evidence would be:

$$M'(A) = M1(A) \oplus M2(A) = 1-(1-M1(A))\bullet(1-M2(A))$$

(where (') represents Mass at the end of Step 1)

Since only evidence confirming (A) is used here, Mass is assigned by M1 or M2 only to (A) or to (THETA), see Figure 4.2. No Mass will be assigned to (Null Set). (1-M1(A)) and (1-M2(A)) represent Mass going to THETA from each individual assignment. Their product, $(1-M1(A))\bullet(1-M2(A))$ represents combined Mass going to THETA. The remainder of Mass, $1-(1-M1(A))\bullet(1-M2(A))$, goes to (A). No normalization is required since no Mass goes to the Null Set.

Suppose two more pieces of evidence are received, M3 and M4, that disconfirm A. Mass from step 1 assigned to (not A) would be:

$$M'(\text{not A}) = 1-(1-M3(\text{not A}))\bullet(1-M4(\text{not A}))$$

$1 - M'(\text{not A})$ = Mass assigned to THETA

This process continues for each element in THETA. The result is $2\bullet N$ Belief functions, where each Belief function has two components one for each element and for each complement of the element.

b. Step 2: Combination of Element Pairs

In Step 2, the Mass for each element and its complement are combined. Figure 4.3 shows an orthogonal sum example of the combination for Mass that supports an item and its complement.

	M2(A)	1-M2(A) or M2(THETA)
M1(A)	(A) M1(A)•M2(A)	(A) M1(A)•(1-M2(A))
1-M1(A) or M1(THETA)	(A) (1-M1(A))•M2(A)	(THETA) (1-M1(A))•(1-M2(A))

Therefore: M(A) = 1-(1-M1(A))•(1-M2(A))
= 1-(Mass going to THETA)

Figure 4.2 Orthogonal Combination For Step 1.

	M'(not A)	1-M'(not A) (THETA)
M'(A)	(Null Set) M'(A)•M'(not A)	(A) M'(A)•(1-M'(not A))
1-M'(A) (THETA)	(not A) (1-M'(A))•M'(not A)	(THETA) (1-M'(A))•(1-M'(not A)

Figure 4.3 Orthogonal Combination For Step 2.

In this step Mass will be assigned to the element, complement of the element, THETA and the Null Set. The Mass going to the Null Set:

$$M''(\text{Null Set}) = M'(A) \cdot M'(\text{not } A)$$

must be removed and so the Masses are normalized by the factor:

$$1 - M'(A) \cdot M(\text{not } A) \quad (1 - \text{Mass going to the Null Set})$$

Mass going to (A) in this step, M"(A), will be the combined Mass of the intersection of M'(A) and 1-M'(not A), the Mass of the complement that was assigned to THETA from Step 1. The same analogy is true for the combined Mass going to (not A) in Step 2, M"(not A), see Figure 4.3.

Now the number of Belief functions has been reduced from 2•N to N, one for each element and complement pair. The resultant Masses are represented by Barnett as p and c, where:

$$p = M''(A) = M'(A) \cdot (1 - M'(\text{not } A)) / (1 - M'(A) \cdot M'(\text{not } A))$$
$$c = M''(\text{not } A) = M'(\text{not } A) \cdot (1 - M'(A)) / (1 - M'(A) \cdot M'(\text{not } A))$$

Also, the Mass assigned to THETA after combination in Step 2 is represented by r:

$$r = 1 - p - c$$

and the variable, d, used later in Step 3 for normalization is the Mass assigned to the complement of the element and to THETA. "d" is used in Step 3 to compute the Plausibility of each element and is identified now for later computations.

$$d = r + c$$

c. Step 3: Combination Within THETA

Now the N Belief functions must be reduced to one by combination within the Frame of Discernment. The example Frame in this chapter: (A,B,C), will be used to demonstrate computations in Step 3. All results from Step 2 will be represented as a function of their element, i.e.:

$$p(A) = M^n(A)$$
$$c(A) = M^n(\text{not } A)$$
$$r(A) = 1 - p(A) - c(A)$$
$$d(A) = r(A) + c(A)$$

The normalizing factor, K, for the overall Belief function is:

$$K = 1/(\text{PROD } d(i)) \cdot (1 + \text{SUM } p(i)/d(i) - \text{PROD } c(i),\ i \text{ in THETA} \quad (4.1)$$

or:

$$K = 1/[d(A) \cdot d(B) \cdot d(C) \cdot (1 + p(A)/d(A) + p(B)/d(B) + p(C)/d(C)) - c(A) \cdot c(B) \cdot c(C)]$$

Belief for each item (i) is then computed by:

$$Bel(i) = K(p(i) \cdot \text{PROD } d(j) + r(i) \cdot \text{PROD } c(j)),\ j \neq i \quad (4.2)$$

then:

$$Bel(A) = K \cdot p(A) \cdot d(B) \cdot d(C) + r(A) \cdot c(B) \cdot c(C)$$

Belief for the complement of the item is computed by:

$$Bel(\text{not}(i)) = K \cdot [(\text{PROD } d(i) \cdot (\text{SUM } p(j)/d(j)) + c(i) \cdot \text{PRODd}(j) - \text{PROD } c(i)],\ j \neq i \quad (4.3)$$

$$Bel(\text{not } A) = K[\,(d(A)\bullet d(B)\bullet d(C))\bullet(p(B)/d(B) + p(C)/d(C)) + c(A)\bullet d(B)\bullet d(C) - c(A)\bullet c(B)\bullet c(C)\,]$$

All Belief is computed in this manner to produce one Belief function. Evidential Intervals are then computed as follows:

$$\text{Evidential Interval } (A) = [Bel(A), Bel(\text{not } A)]$$

The following numerical solution of the entire method using the example will be presented to aid in the understanding of this method.

Eight items of evidence are received and Masses are assigned as follows:

Item 1 $M(A) = 0.6$
Item 2 $M(\text{not } B) = 0.3$
Item 3 $M(C) = 0.4$
Item 4 $M(B) = 0.8$
Item 5 $M(\text{not } A) = 0.5$
Item 6 $M(\text{not } C) = 0.2$
Item 7 $M(C) = 0.7$
Item 8 $M(C) = 0.1$

Step 1: combine evidence for each item and each complement. Only (C) has multiple evidence and is combined as:

$$M'(C) = 1-(1-M1(C))\bullet(1-M2(C))\bullet(1-M3(C))$$
$$M'(C) = 1 - (1 - 0.4)\bullet(1 - 0.7)\bullet(1 - 0.1) = 0.838$$

Step 2: combine the element and its complement:

$$p(A) = M''(A) = M'(A)\bullet(1-M'(\text{not } A))/(1-M'(A)\bullet M'(\text{not } A))$$
$$p(A) = M''(A) = (0.6)\bullet(1-0.5)/(1-(0.6)\bullet(0.5)) = 0.429$$

$$c(A) = M''(\text{not } A) = M'(\text{not } A)\bullet(1-M'(A))/(1-M'(A)\bullet M'(\text{not } A))$$
$$c(A) = (0.5)\bullet(1-0.6)/(1-(0.6)\bullet(0.5)) = 0.286$$

$r(A) = 1 - p(A) - c(A)$

$r(A) = 1 - 0.429 - 0.286 = 0.285$

$d(A) = r(A) + c(A)$

$d(A) = 0.285 + 0.286 = 0.571$

$p(B) = (0.8) \cdot (1-0.3)/(1-(0.8) \cdot (0.3)) = 0.737$

$c(B) = (0.3) \cdot (1-0.8)/(1-(0.8) \cdot (0.3)) = 0.079$

$r(B) = 1 - 0.737 - 0.079 = 0.184$

$d(B) = 0.184 + 0.079 = 0.263$

$p(C) = (0.838) \cdot (1-0.2)/(1-(0.838) \cdot (0.2)) = 0.805$

$c(C) = (0.2) \cdot (1-0.838)/(1-(0.838) \cdot (0.2)) = 0.039$

$r(C) = 1 - 0.805 - 0.039 = 0.156$

$d(C) = 0.156 + 0.039 = 0.195$

Step 3:

$K = 1/[\ (.571) \cdot (.263) \cdot (.195) \cdot (1 + .429/.571 + .737/.263 + .805/.195) - (.286) \cdot (.079) \cdot (.039)\ ] = 8.998$

Computation of Beliefs:

$Bel(A) = 3.988 \cdot (.429 \cdot .263 \cdot .195 + .285 \cdot .079 \cdot .039) = .091$

$Bel(B) = 3.988 \cdot (.737 \cdot .571 \cdot .195 + .184 \cdot .286 \cdot .156) = .360$

$Bel(C) = 3.988 \cdot (.805 \cdot .571 \cdot .263 + .156 \cdot .286 \cdot .079) = .496$

$Bel(\text{not } A) = 3.988\,(.029 \cdot (.737/.263 + .805/.195) + .286 \cdot .263 \cdot .195 - .001) = .856$

$Bel(\text{not } B) = 3.988 \cdot (.029 \cdot (.429/.571 + .805/.195) + .014) = .620$

$Bel(\text{not } C) = 3.988 \cdot (.029 \cdot (.429/.571 + .737/.263) + .014) = .467$

The Evidential Intervals would then be:

$A: [\,0.091, 1-0.856\,] = [\,0.091, 0.144\,]$

$B: [\,0.360, 0.380\,]$

$C: [\,0.496, 0.533\,]$

2. Computational Time For Barnett's Method

A program was written to assist in the computations of Barnett's method. The program also enabled comparison of computational time and memory requirements with the normal Dempster-Shafer method. A program listing for Barnett's method is contained in Appendix B. This program was written as a research tool and does not have the user friendly enhancements of the Dempster-Shafer decision aid in Appendix A.

Figure 4.4 shows the timed computations for the Barnett method. The computer memory (320K RAM) could store a Frame of Discernment of up to 1000 items compared to 9 for full Dempster-Shafer. Figure 4.5 compares the Belief calculation times for Barnett with the times already shown for the Dempster-Shafer aid.

Barnett's method offers a very appealing and time efficient use of Dempster-Shafer in a system where the following criteria are met:

a) The Frame of Discernment can be adequately represented by the singleton elements and their complements.

b) All evidence can be divided into confirming and disconfirming categories for each hypothesis.

The example of the avenues of approach scenario could be well represented by this method. As shown, evidence based on terrain, weather, obstacles, etc., could be combined using Barnett's method. The resultant Evidential Intervals would then be compared to determine the most likely enemy route, or the best friendly route.

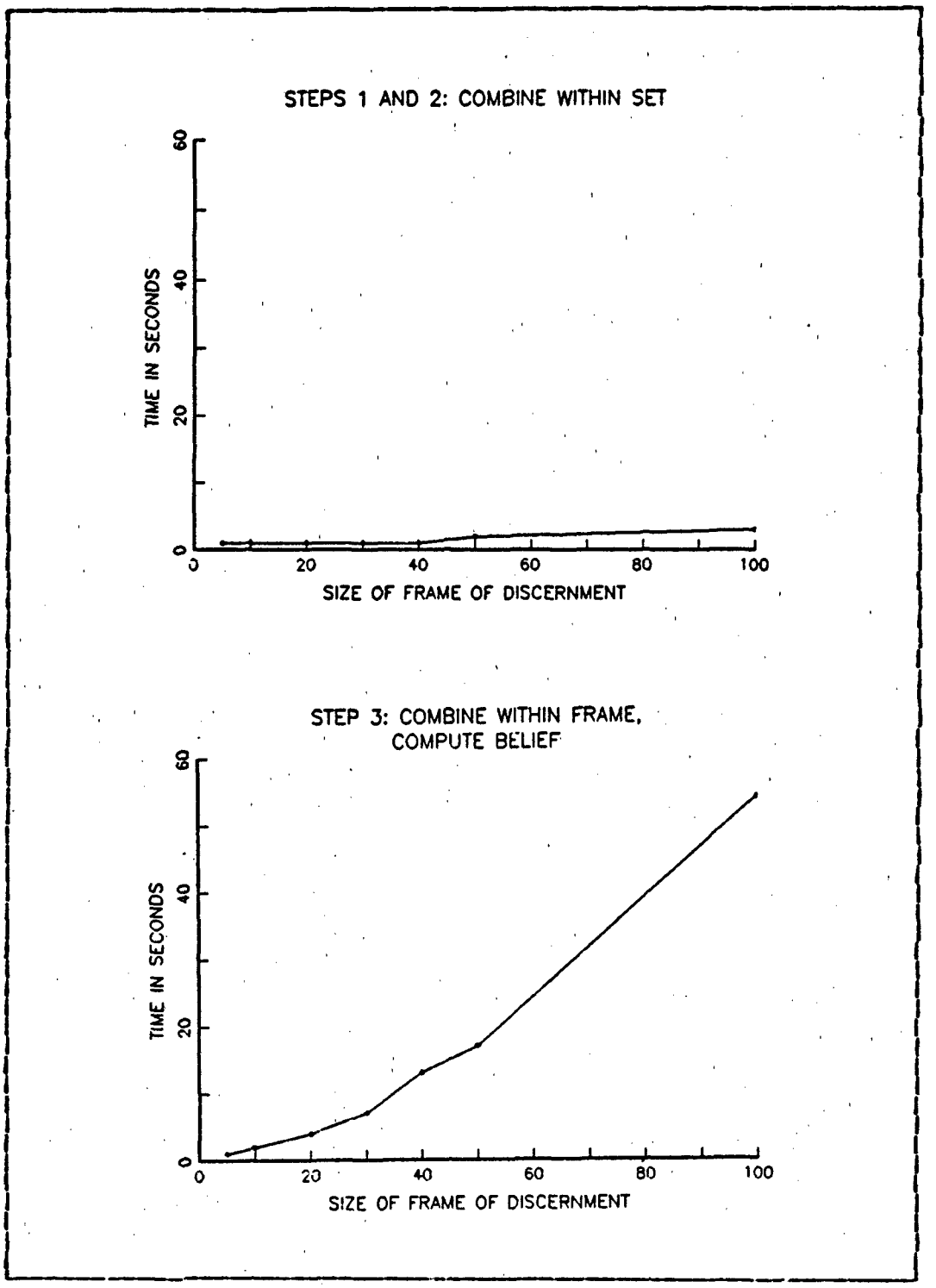

Figure 4.4 Computation Times For Barnett's Method.

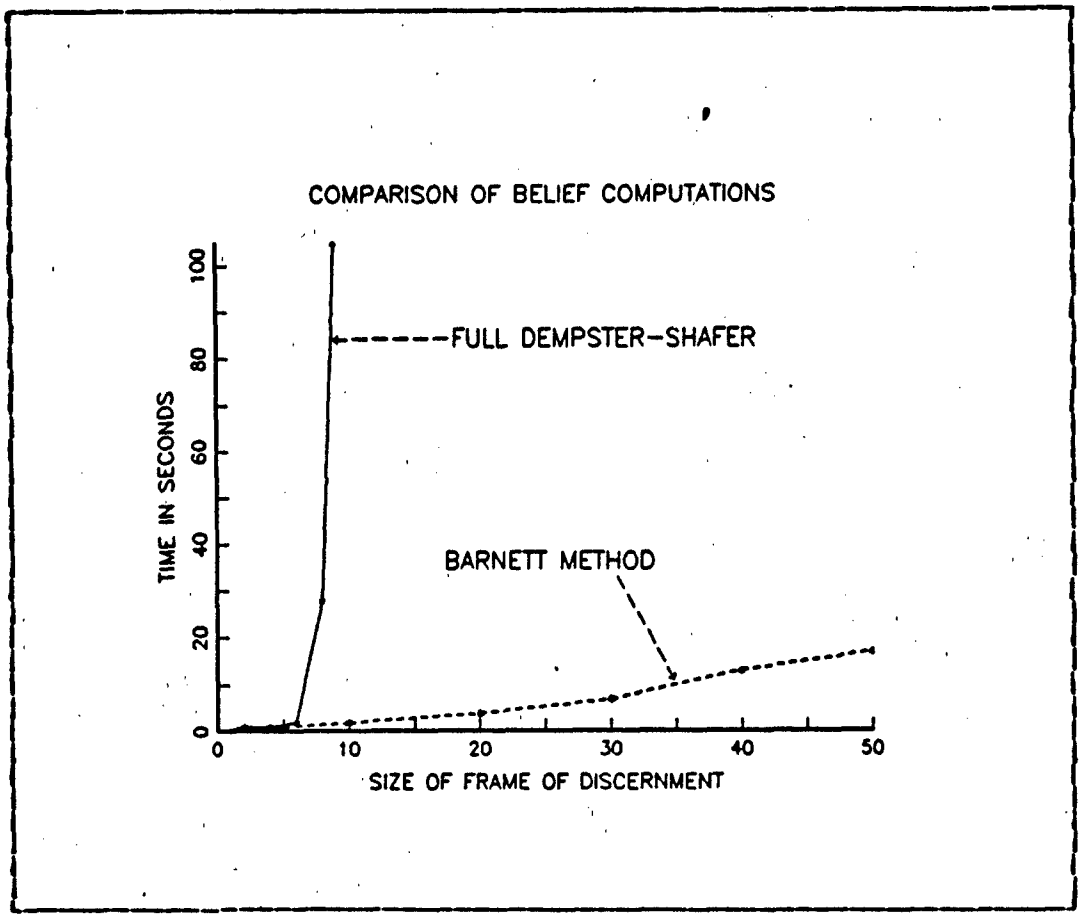

Figure 4.5 Comparison Of Barnett With Full Dempster-Shafer.

C. GORDON AND SHORTLIFFE METHOD, HIERARCHICAL HYPOTHESIS SPACE

Gordon and Shortliffe [Ref. 15] agree that if evidence confirms or disconfirms singleton hypotheses, then Barnett's method produces the desired time reduction. However, there are many classes of problems where more flexibility is required. If the hypothesis space can be reduced to a strict hierarchy, many more real world problems could be handled. Gordon and Shortliffe use this hierarchical approach in their work with MYCIN, a medical diagnostic aid,

but it is also well suited to a military example where enemy intentions are being determined.

The example of enemy attack can be expanded so that the analyst has to consider the overall enemy intention: attack, reinforce, defend, delay, or withdraw. These intentions can be structured into a hierarchical tree such as that in Figure 4.6. Grouped in this manner, the example now fits the method proposed by Gordon and Shortliffe.

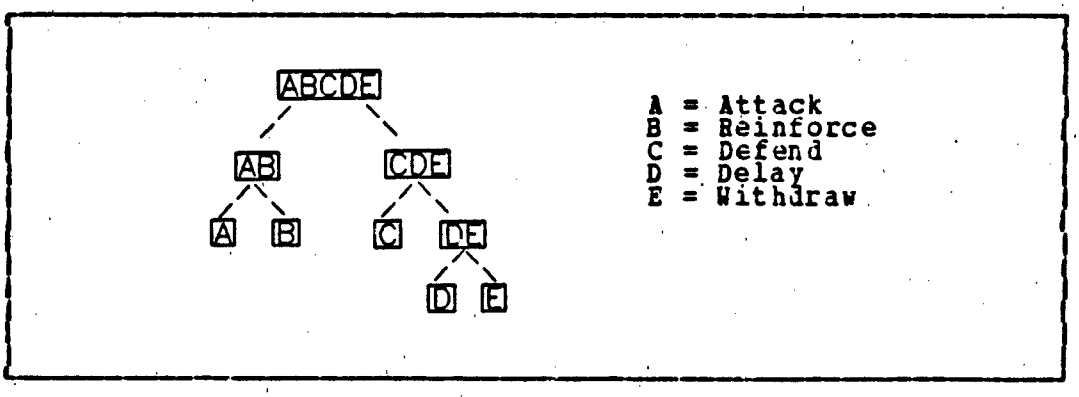

Figure 4.6 Enemy Intention Hierarchical Tree.

The enemy intentions are divided into two main groups, aggressive (attack,reinforce), or regressive (defend, delay, withdraw). The regressive intentions are further divided into stationary (defend) and motionary (delay, withdraw) intentions. Each element or subject in the tree has only one parent, for a strict hierarchy. Since the enemy is capable of only one major tactic in the zone of consideration, there is no interest in the pair (attack,defend), or other such combinations that have no meaning to the analyst. Also, the analyst must have the capability to separate sensor data into support for these elements of the hierarchical tree. The evidence received will only apply to these elements in the tree and their complements.

1. Formulation Of Gordon and Shortliffe's Method

This hierarchical approach proceeds similarly to the singleton hypotheses method. All evidence is divided into confirming or disconfirming parts, only now there are pairs, triplets, etc. to which this evidence may be assigned. As in Barnett's method, three steps will be used in the combination of evidence:

a) Combine evidence for each element and each element complement.
   i) Combine all confirming evidence. Same as Barnett's first step.
   ii) Combine all disconfirming evidence
b) Combine all confirming evidence in THETA.
c) Combine disconfirming evidence with confirming evidence from Step 2.

Before proceeding, the hierarchical tree of Figure 4.6 will be split into a tree with the elements of THETA, T, and one with the complements of T, Tc, see Figure 4.7 Since it is the superset of both trees, (THETA) itself, is not included in either tree.

T = (AB,CDE,DE,A,B,C,D,E)
Tc = (notAB,notCDE,notDE,notA,notB,notC,notD,notE)

Evidence will be combined for |T| + |Tc| items = 16, and final belief will be computed after Step 3 for |T| + THETA = 17 items versus 2 exp 5 = 32 items for the full Dempster-Shafer method. These reductions will decrease computational times and storage requirements.

The steps for Gordon and Shortliffe's Method are as follows.

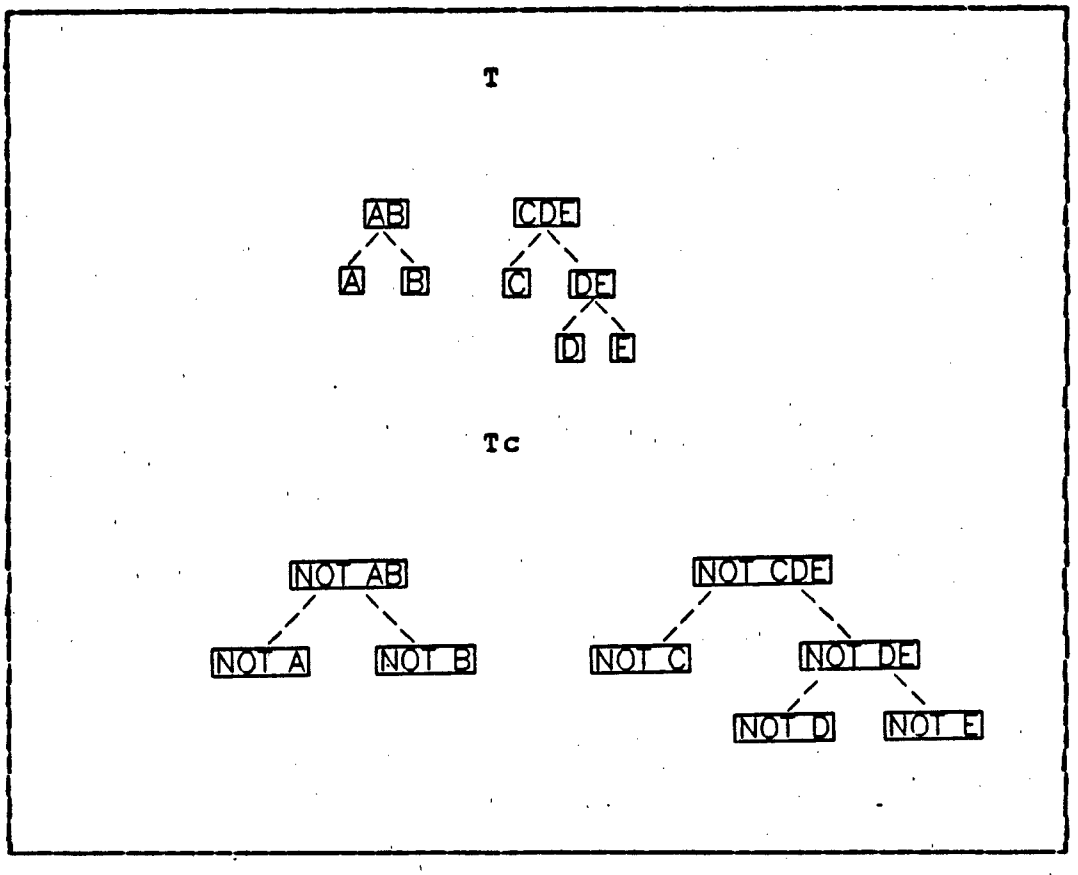

Figure 4.7 Hierarchical Tree Of Elements And Complements.

a. Step 1: Combine Evidence For Elements And For Complements

Use Barnett's equations to compute combined evidence for each element and each complement:

$$M'(i) = 1-(1-M1(i))(1-M2(i)),\ i \text{ in THETA} \qquad (4.4)$$

Where (') indicates Mass after Step 1 and (") will indicate Mass after Step 2.

$$M'(A) = 1-(1-M1(A)) \bullet (1-M2(A))$$

Each element (i) will now have a Mass M'(i) and a Mass 1-M'(i) assigned to THETA. It is necessary to identify the Mass that each element's combination sends to THETA for use in later steps. Let M'i(THETA) equal the Mass assigned to THETA by element (i) during step 1.

M'A(THETA) = 1-M'(A)

The Masses for (not A) would then be:

M'(not A) = 1-(1-M1(not A)) • (1-M2(not A))
M'notA(THETA) = 1-M'(not A)

There are now 2•N Belief functions, N in T and N in Tc. Step 2 will reduce the N Belief functions in T to one Belief function.

b. Step 2: Combine All Confirming Evidence in THETA.

All confirming evidence in THETA, all of which is in T, will now be combined.

The combined evidence for THETA, M"(THETA) is:

$$M''(THETA) = K \bullet PROD\ M'i(THETA),\ i \text{ in } T \qquad (4.5)$$

This is the product of all Mass assigned to THETA by all elements in T. Now compute the combined evidence for all other elements in T:

$$M''(i) = K \bullet M'(i) \bullet PROD\ M'j(THETA),\ i \text{ in } T,\ j \text{ not a superset of } i \qquad (4.6)$$

The Mass of element (i) in Step 2 is the product of its Mass with the Mass assigned to THETA by all other sets in T except (i), and supersets of (i). Element (i) cannot

combine with Mass that it assigned to THETA. In addition, it cannot receive Mass from a superset's assignment to THETA. The evidence that was assigned to a superset of (i) was not conclusive enough to assign it to element (i). Allowing (i) to combine with M'j(THETA), (j) a superset of (i), would allow assignment of Mass from which (i) was earlier restricted. This is not acceptable under Dempster-Shafer. The Mass for element (A) in the example would be:

$$M''(A) = K \bullet M'(A) \bullet M'B(THETA) \bullet M'CDE(THETA) \bullet M'C(THETA) \bullet M'DE(THETA) \bullet M'D(THETA) \bullet M'E(THETA)$$

These calculations continue for all elements in T. The normalization factor, K, is the inverse of the sum of all the new Masses:

$$K = 1/\text{SUM } M''(i),\ i \text{ in } T \text{ or } i = THETA \qquad (4.7)$$

There are now N+1 Belief functions, N in Tc and one in T. Step 3 will reduce these to one Belief function.

c. Step 3: Combine Disconfirming And Confirming Evidence

Step 3 will now combine T and Tc to produce one Belief function. Each element of Tc will combine with T. Step 3 uses an approximation to ∋, the orthogonal sum, which will be designated as ∋'. M"(i) ∋' M"(not i) will have non zero value on only (T union THETA), (ABCDE,AB,CDE,DE,A,B,C,D,E). Any Belief normally assigned by ∋ to q, q not in T, will instead, by ∋', be assigned to the first ancestor of q in T. For example:

(not E) ∋' (CDE) = (ABCD) ∋ (CDE)
but (ABCD) & (CDE) = (CD), which is not in T.

In this case the Mass that would have been assigned to (CD) will be assigned to (CDE) the first ancestor in T.

There are three cases for step 3 combinations. For each case there are two elements involved, one from Tc, (not Y), and one from T, (X). The combination case is based on the relationship of (Y) and (X).

1. Case 1: X is a subset of Y:

   M(X) = K•M"(X)•M"notY(THETA)

   For example:

   M"(A) ∂' M"(not AB), A is a subset of AB
   M(A) = K•M"(A)•M"notAB(THETA)

2. Case 2: X & Y = (Null Set), i.e. X & notY = X

   a) Case 2(a): If (X union Y) in (T union THETA):
      M(X) = K[M"(X) + M"(X union Y)•M"(not Y)]

      For example:
      M" ∂' M"(not DE), C & DE = (Null Set),
      CDE in (T union THETA), therefore:
      M(C) = K•[M"(C) + M"(CDE)•M"(not DE)]

   b) Case 2(b): If (X union Y) not in (T union THETA) then the Mass of (X) is not changed by the combination:
      M(X) = K•M(X)

      M"(C) ∂' M"(not E), C & E = Null Set,
      CE not in (T union THETA), therefore:
      M(C) = K•M"(C)

3. Case 3: X is a proper super set of Y

   a) Case 3(a): If X & notY is a set in T:
      M(X) = K•M"(X)•M"notY(THETA)

M"(CDE) @' M"(notC), CDE & ABDE = DE, a set in (T union THETA)

M(CDE) = K•M"(CDE)•MnotC(THETA)

b) Case 3(b): If X & notY is not a set in T (this case assigns Mass to the superset, which in all cases is X due to the strict hierarchy and unique parent requirements).

M"(notE) @' M"(CDE), ABCD & CDE = CD, not in (T union THETA), but CDE is the first ancestor of CD

M(CDE) = K•M(CDE)

This process continues with each element from Tc combining with each element in T using one of the case rules. K, the normalizing factor is computed after each iteration of (not Y) from Tc.

Normalization is done by summing the Masses in (T union THETA) and dividing all Masses by that value.

$$K = 1/\text{SUM } M(i),\ i \text{ in } (T \text{ union } THETA) \qquad (4.8)$$

A shortcoming of the method described by Gordon and Shortliffe is that step 3 assigns all Mass to (T union THETA). No Mass remains in the complement sets, therefore it is not possible to compute Evidential Intervals, [Bel(A), 1-Bel(not A)]. All comparisons between hypotheses must be done on Belief alone.

The following example of this method is provided for clarification. The hierarchical tree shown earlier in Figure 4.4 still applies.

Masses are assigned as follows:

M(AB) = 0.4,	M AB(THETA) = 0.6
M(not CDE) = 0.3,	M notCDE(THETA) = 0.7
M(DE) = 0.6,	M DE(THETA) = 0.4
M(A) = 0.4,	M A(THETA) = 0.6

M(C) = 0.2, M C(THETA) = 0.8

M(A) = 0.3, M A(THETA) = 0.7

There is only one element in Tc = (not CDE), there are four in T.

Step 1:

(A) is the only element with multiple Masses:

M'(A) = 1-(M1(A))•(1-M2(A))

M'(A) = 1-(1-0.4)•(1-0.3) = 0.58

M'A(THETA) = 1-0.58 = 0.42

Step 2: Combine all confirming evidence in T.

M"(THETA) = K•M'ABC(THETA)•M'DE(THETA)
•M'A(THETA)•M'C(THETA)
= K•(0.9)•(0.5)•(0.42)•(0.8) = K•(0.081)

M"(AB) = K•M'(AB)•M'A(THETA)•M'DE(THETA)•M'C(THETA)
= K•(0.1)•(0.42)•(0.5)•(0.8) = K•(0.017)

M"(DE) = K•M'(DE)•M'A(THETA)•M'AB(THETA)
= K•(0.5)•(0.42)•(0.9) = K•(0.189)

M"(A) = K•M'(A)•M'DE(THETA)
= K•(0.58)•(0.5) = K•(0.29)

M"(C) = K•M'(C)•M'AB(THETA)•M'A(THETA)•M'DE(THETA)
= K•(0.2)•(0.9)•(0.42)•(0.5) = K•(.038)

K = 1/The sum of all Masses in (T union THETA)
= 1/M"(THETA) + M"(AB) + M"(DE) + M"(A) + M"(C)
= 1/(0.151) + (0.017) + (0.189) + (0.29) + (0.038)
= 1/0.685 = 1.46

Normalize the masses in T.

The elements in T are:

M"(THETA) = (1.46)•(.151) = .221
M"(AB) = (1.46)•(.017) = .025
M"(DE) = (1.46)•(.189) = .276
M"(A) = (1.46)•(.29) = .423
M"(C) = (1.46)•(.038) = .055
Total = 1.0

The elements in Tc are:

M"(not CDE) = 0.3 M"notCDE(THETA) = 0.7

Step 3:

1. M"(notCDE) @' M"(THETA), THETA a super set of AB => Case 3

   THETA & (not CDE) = THETA & AB = AB in (T union THETA)

   M(THETA) = K•M"(THETA)•M"notCDE(THETA) In this case, Mass would normally go to (THETA) and AB, see Figure 4.2, here (THETA) its combined Mass for this iteration, and later when M"(not CDE) @' M"(AB) occurs, AB will receive its combined Mass. This is an iterative process versus the normal one step Dempster-Shafer combination.

   M(THETA) = (.221)•(.7) = 0.155(normalization will cccur later)

2. M"(not CDE) @' M"(AB), ABCDE = Null Set, (AB union CDE) in (T union THETA) => Case 2

   M(AB) = [M(AB) + M(ABCDE)•M(not CDE)]
   = [.025 + (.221)•(.3)] = .091

3. M"(not CDE) @' M"(DE), DE subset of CDE => Case 1

   M(DE) = M"(DE)•MnotCDE(THETA)
   = (.276)•(.7) = .193

4. M(not CDE) @' M"(A), CDE^A = Null Set => Case 2, (A union CDE) not in (T union THETA)

   M(A) = M(A) (but will be normalized)

   = .423

5. M"(not CDE) @' M"(C), C a subset of CDE => Case 1

   M(C) = M"(C) •M"notCDE(THETA)

   = (.055) • (.7) = .039

Compute K, the normalization factor.

K = 1/ sum of Masses in (T union THETA)

= .155 + .091 + .193 + .423 + .039 = .901

Normalized Masses:

M(THETA) = .155/.901 = .172

M(AB) = .091/.901 = .101

M(DE) = .193/.901 = .214

M(A) = .423/.901 = .470

M(C) = .039/.901 = .043

Total = 1.0

This process of combining elements from Tc with those in T would continue until all elements have been combined, in this example there was only one. Comparison of hypotheses would then be done based on final Belief values:

Bel(AB) = M(AB) + M(A) = .101 + .470 = .571

Bel(DE) = M(DE) = .214

Bel(A) = .470

Bel(C) = .043

Bel(B) = Bel(D) = Bel(E) = 0.0

This would show strongest Belief (.571) in an aggressive enemy action (AB), which stands for attack or reinforce.

2. Computational Time For The Gordon and Shortliffe Method

A program was written to assist in the computations using the Gordon and Shortliffe technigue. A listing of the program is contained in Appendix B. This program, like the program for Barnett's method, does not have the user friendly enhancements of the Dempster-Shafer decision aid in Appendix A. The purpose of the program was to assess the time calculation advantage of this method and determine memory requirements.

Figure 4.8 shows a comparison of the Belief computations for the three methods discussed so far. Belief computations for Gordon and Shortliffe are the same as the full Dempster-Shafer, except there are fewer sets for which computations are necessary. The time reduction occurs since many of the subsets of THETA are not considered.

The maximum number of subsets that can occur under the unique parent restriction of the method are: (2•N - 1). This number results from a descending creation path from THETA, separating one element at a time. Now instead of having 2 exp N elements for which Belief must be computed, there are only (2•N-1). For a THETA of 10 elements, instead of having 2 exp 10 = 1024 elements, there may only be a maximum of 2•10 - 1 = 19 elements. This size reduction helps explain the computation times in Figure 4.8.

The computer memory (320K RAM) could store up to a Frame of Discernment of up to 500 items using the Gordon and Shortliffe method versus 1000 for Barnett and 9 for the full Dempster-Shafer.

Gordon and Shortliffe's method is a practical and efficient use of Dempster-Shafer in a system where the following criteria are met:

a) A strict hierarchy of elements exists and each element has only one parent.

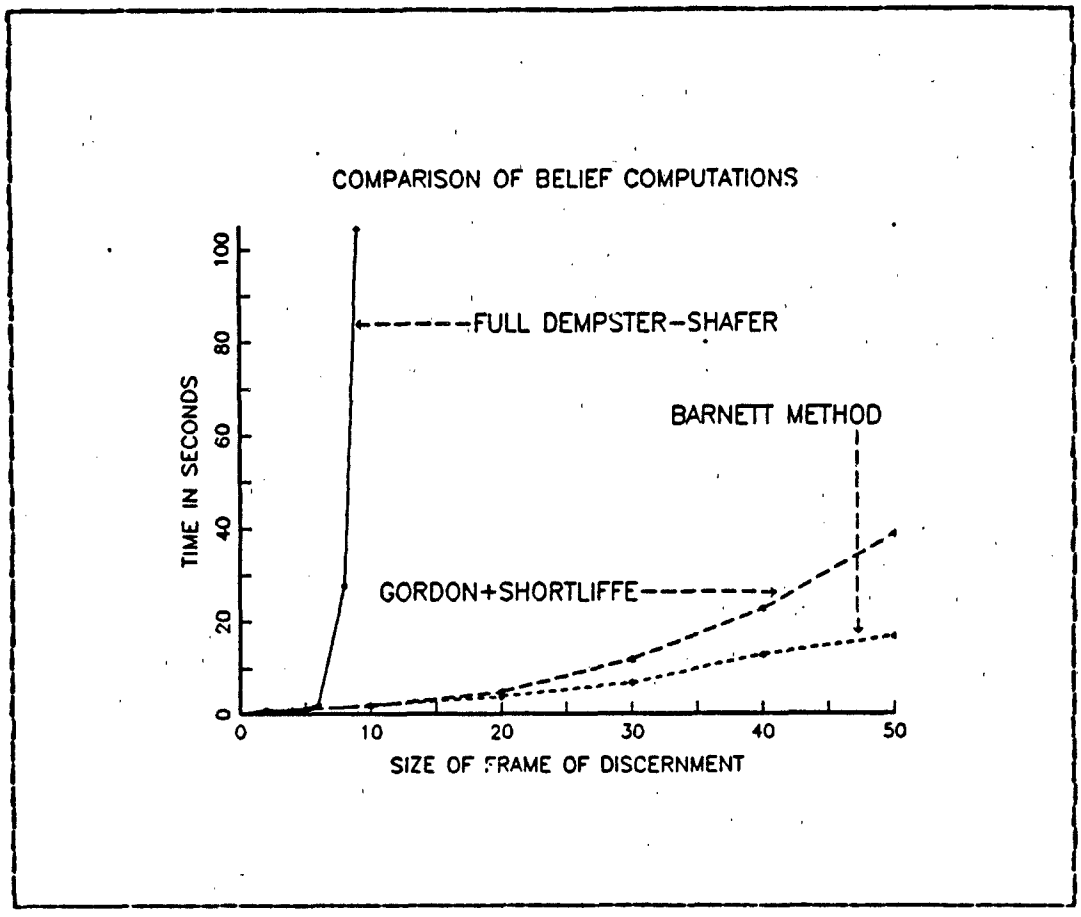

Figure 4.8 Comparison Of Belief Computational Times.

b) All evidence can be divided into confirming and disconfirming categories for each hypothesis.

The example of the overall enemy intention fits well within the limits of this method. However, the comparison of the hypotheses of the intentions would have to be made on the result of Belief alone. The analyst would not know the Ignorance, 1-Bel(not X), remaining about each of the hypothesis.

Shortcomings of this method are:

a) Loss of ability to compute Evidential Intervals as discussed earlier.

b) Order dependence on combination. Gordon and Shortliffe mention that their approximation to the Dempster-Shafer combination can be order dependent when a set and its parent have only one descendent.

D. MULTIPLE FRAMES OF DISCERNMENT

One final method of reducing complexity, multiple Frames of Discernment, will now be discussed. This method may use any of the computational techniques discussed so far. Its reduction in complexity comes from the separation of the Frame into smaller, more manageable categories with which the analyst can work. Computation of Belief for these smaller Frames should now fall into the reasonable area of Dempster-Shafer calculation times.

The obvious requirement for this method is a logical separation of elements in the original Frame of Discernment. Once the elements are separated into multiple Frames, items from different Frames cannot be compared for they are now part of different Belief functions.

As an example of this method, consider an expanded version of the enemy intentions problem. The analyst still desires to determine the overall intention. If the intention is Attack, he wants to know which sector is most likely. He also desires to know if the enemy intends to use nuclear weapons, no matter what the tactic. The Frame of Discernment now has 9 items:

1. Attack, Sector 1
2. Attack, Sector 2
3. Attack, Sector 3
4. Reinforce
5. Defend
6. Delay
7. Withdraw

8. Use Nuclear Weapons

9. Not Use Nuclear Weapon

A hierarchy of elements exists, but as described in Chapter III with the sector example, unique parents do not exist. For example, let the Frame of Discernment be represented as: A,B,C,D,E,F,G,H,I where:

A = Attack, Sector 1

B = Attack, Sector 2

C = Attack, Sector 3

D = Reinforce

E = Defend

F = Delay

G = Withdraw

H = Use Nuclear Weapons

I = Not Use Nuclear Weapons

Now (ABC) is a multiple unique parent for (AB), (BC) and (AC), but (A) has two parents (AC) and (AB). The same situation exists for (B) and (C). Therefore, Gordon and Shorliffe's method would not work here.

The full Dempster-Shafer method would create 2 exp 9 subsets, which according to Figure 4.1 would require about 100 seconds for Belief computations. Some of the pairs of elements of THETA, as discussed earlier in the Gordon and Shortliffe method, would not be of interest to the analyst. A solution to this dilemma is separating the problem into three Frames of Discernment. Each Frame will use one of the computational methods discussed. Any of the methods may be used where applicable in the multiple frames. All three are used here to show the diversity of this method.

The overall intention of the enemy is still desired, so use the Gordon and Shortliffe method for the Frame: (AT,D,E,F,G), where (AT) is the attack intention and D through G remain as described above.

Use Barnett's method for the question of the enemy's intent to use nuclear weapons, (H,I), and use the full Dempster-Shafer method to determine sector of attack. Caution must be exercised when discussing the most likely sector of attack for that Belief is conditioned on the Belief that the overall intent is attack.

The timed calculations would now be:

Full Dempster-Shafer: 3 elements in THETA = approximately 1 second

Gordon and Shortliffe: 5 items in THETA = approximately 1 second

Barnett: 2 itmes in THETA = approximately 1 second

This complex and inefficient problem has now been reduced to a very manageable calculation for the intelligence analyst using Dempster-Shafer.

Evidence received while using this method of multiple Frames of Discernment, does not need to be separated into one of the multiple Frames. On the contrary, a report indicating an attack with nuclear weapons could be used for all the Frames if applicable. The Belief values of the various Frames will not be compared and are not calculated using the Masses of the other Frames.

Intermediate results can be saved on a disk and recalled when more relevant data arrives. Therefore, one machine could keep all three methods running in an almost simultaneous state.

The method of Multiple Frames of Discernment is a viable alternative to the full Dempster-Shafer method if the Frame is separable into distinct categories.

V. SITUATION DEVELOPMENT ANALYSIS, AN APPLICATION AREA FOR DEMPSTER-SHAFER

In Chapter II, the battlefield intelligence process was modelled as the Interpretation node in the command and control cycle. In this node, information was processed into intelligence by the analyst. Chapter III proposed the Dempster-Shafer theory of evidence combination as an aid for the analyst in the Interpretation node. A decision aid and three computational techniques for reducing calculation time for the Dempster-Shafer theory were presented in Chapter IV.

Enhancement of a specific job application in the Intelligence node, through the use of the Dempster-Shafer theory and a decision aid similar to the one in Chapter IV, will now be analyzed. This specific job area, Army Division Situation Development Analyst (DSDA), was chosen due to its relevance to research conducted by MAJ L. Baltezore, U.S. Army, in his thesis at the Naval Postgraduate School [Ref. 16].

Baltezore's thesis proposed a Decision Support System hardware layout to assist the analyst in conducting situation assessment at division level. A Knowledge Based System (KBS) was designed to conduct automated analysis concerning possible courses of enemy attack. The use of Dempster-Shafer in this KBS will now be explored. While not evaluating any specific technique, Baltezore proposed that some method of inference should be used to aggregate battlefield information stored in the data base. The Knowledge Based System of the Division Situation Development Analyst and an intention assessment capability using the speed of Dempster-Shafer with small Frames of Discernment (Quick Assessment Capability), will be used as specific examples of

actual Interpretation node duties that can be enhanced by Dempster-Shafer.

The jobs of the DSDA will be briefly discussed as will the structure of the KBS. The use of a Dempster-Shafer theory in the Knowledge Based System as well as a Quick Assessment Capability through a decision aid will be evaluated.

A. DIVISION SITUATION DEVELOPMENT ANALYSIS

Situation Analysis at the division level is performed within the All Source Production Section (ASPS) of the All Source Analysis System (ASAS), see Figure 5.1, [Ref. 16: p.25]. The ASAS is an Army project to establish a system that maximizes the productivity of Intelligence and Electronic Warfare (IEW). Through enhanced productivity, the Interpretation node will process information into intelligence more quickly, speeding up the command and control cycle.

The Division Situation Development Analyst is usually the senior intelligence analyst in the All Source Production Section. Using a broad view of the enemy forces, the DSDA must determine key enemy objectives, rank potential enemy courses of action, and identify key targets, command and control nodes, or events indicating a specific course of action. The analyst bases his assessment of the enemy situation on data passed through his work station [Ref. 16: p. 32].

As described in Chapter III, this data is inherently uncertain and incomplete. Using this data, the analyst must interpret as much as possible about the enemy intent as quickly as possible. Figure 5.2 depicts the analyst's production cycle. The analyst would use the proposed Knowledge Based System to assist him in determining the

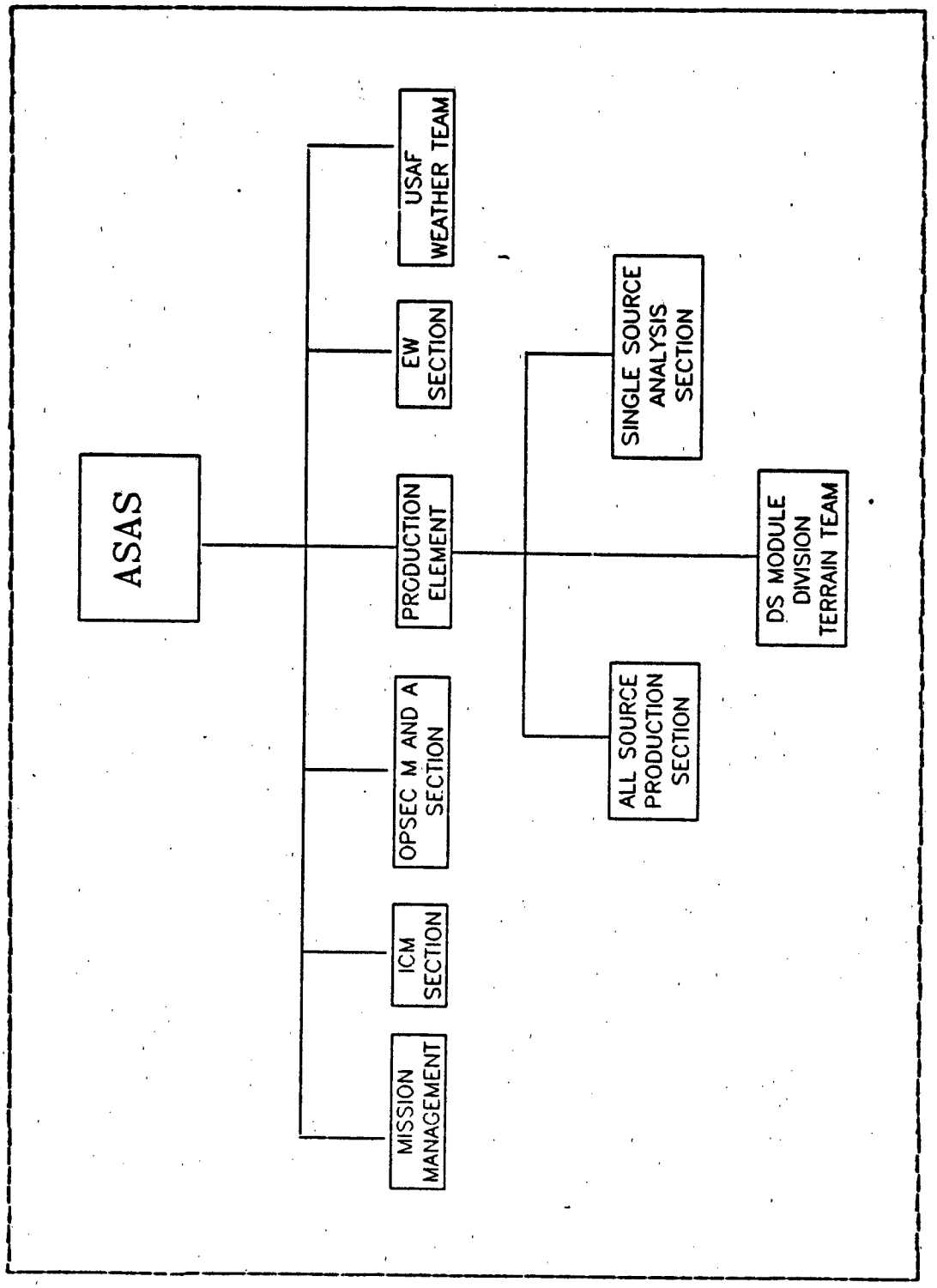

Figure 5.1 ASAS Internal Organization.

possible courses of enemy action. Within this structure exists the potential use of Dempster-Shafer.

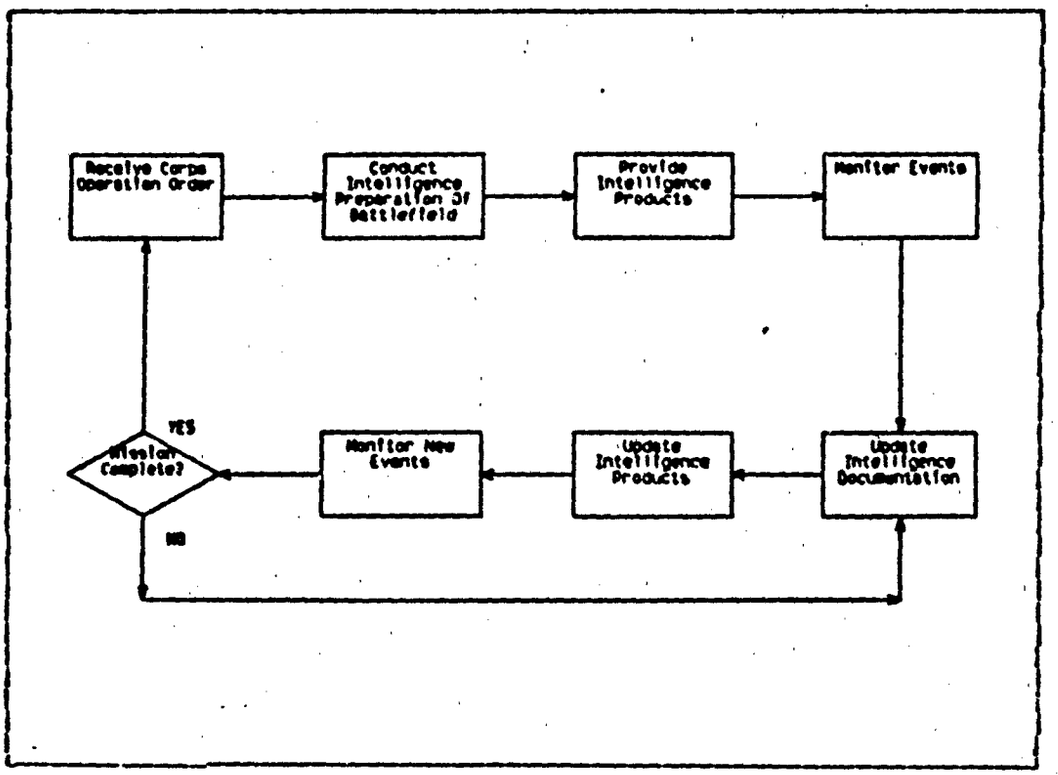

Figure 5.2 Analyst's Production Cycle.

B. THE KNOWLEDGE BASED SYSTEM

As previously stated, the Knowledge Based System is designed to provide automated analysis concerning possible courses of enemy action. Analysis of the KBS will be limited to the structure of the system and the internal theory which correlates the data.

1. Knowledge Based System Structure

Baltezore's proposed structure for the KBS is shown in Figure 5.3. This KBS interacts with the analyst through the Interrogation Module to determine the type of analysis desired. For example, is the enemy intention Attack, (one of the examples used in Chapter IV)? The Situation Assessment Processor accesses the Knowledge Base to determine a rule list associated with the type of analysis desired. Using all data available, the KBS then establishes a probability value associated with the specified course of action. The Explanation Generator presents the analyst with the course of action considered, its probability value, and a rule audit trail of the deduction process.

2. Knowledge Based System Theory

The KBS deduction theory is based on work done by Ben-Basset and Freely, [Ref. 17], who proposed the use of classes, features, and relevancy pointers to conduct situation assessment.

a. Classes

Classes are used to define battlefield situations of interest to the analyst. For example, attack, reinforce, defend, delay, withdraw are enemy situations that may be represented as classes. If the general class is known, such as attack, then the specific location, such as the sector of attack, is desired through analysis. This characteristic is analogous to the Dempster-Shafer frame of discernment.

b. Features

Features are bits of information, such as the context of a report, related to the situation and used to

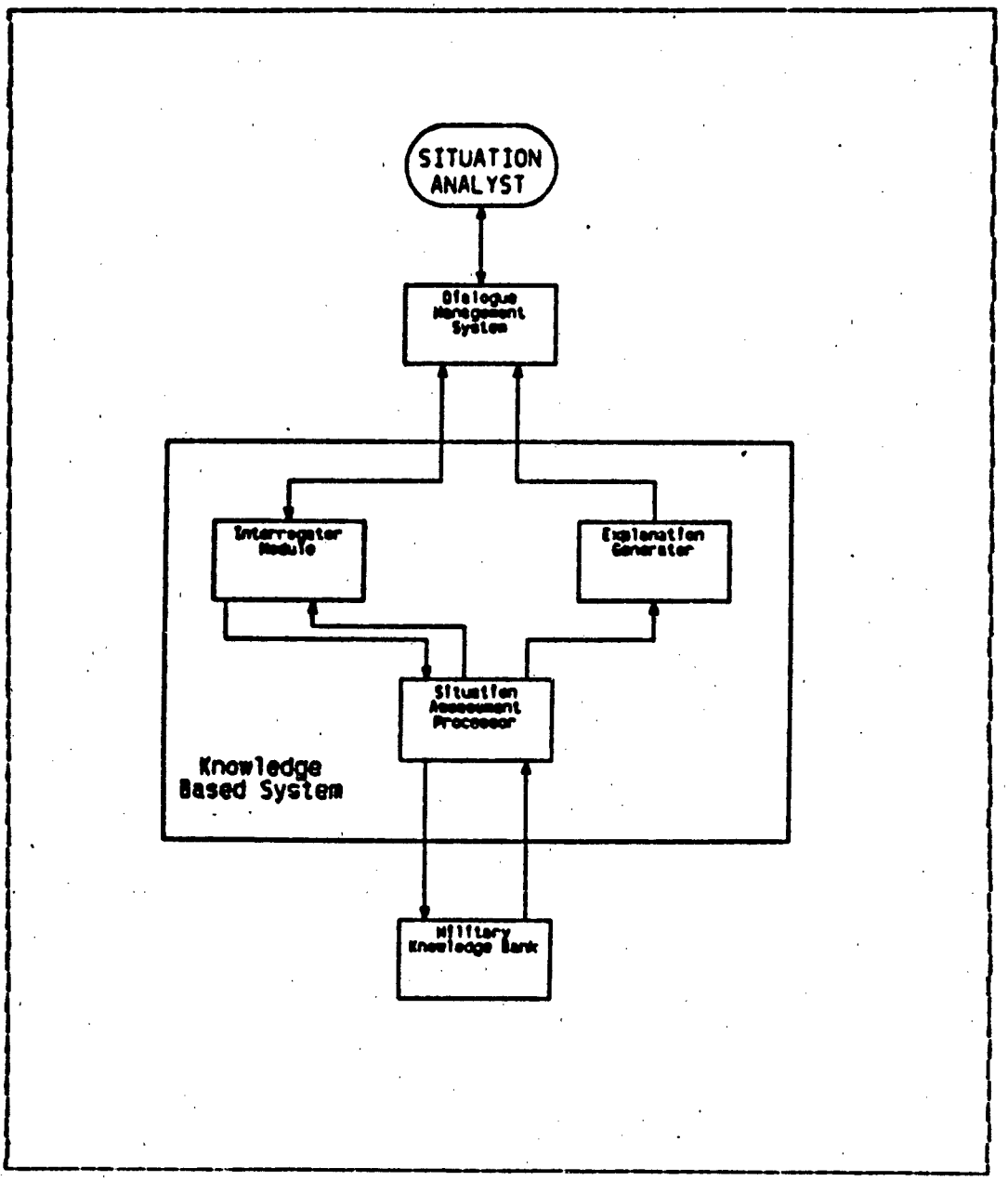

Figure 5.3 Knowledge Based System Structure.

determine the class. For instance, presence of an independent tank battalion (ITB) in the division zone would support the class, attack. An independent tank battalion is a

second echelon unit normally used as a lead attack element in a breakthrough attempt. An example of features and classes is shcwn in Table 5.4.

Class of Enemy Intention: Attack

Features	P	not P
Massing of mechanized elements	0.8	0.3
Extensive artillery preparation	0.8	0.3
Artillery positions concentrated	0.8	0.2
Concentration of mass toward either or both flanks	0.7	0.3
Location of enemy troops in forward assembly area	0.8	0.3
Location of suppl and evac-uation installation well forward	0.7	0.3
Increased air reconnaissance	0.8	0.4
Movement of additional trocps toward the front	0.8	0.4

Figure 5.4 Feature Probability Support For Class.

c. Relevancy Pointers

Relevancy Pointers are used to reduce the expert systems search for features supporting classes. Separate features such as the number and type of tanks may be used to determine the presence of the ITB. If the unit has already been identified, then it is not necessary to use the rules which determine the type of unit. In this manner relevancy pointers speed the assessment process. Relevancy pointers are similar to antecedent rules in a rule-based system.

The significance of features and their support for the different classes under consideration are aggregated by some theory of evidence combination. Features, as information, may support several classes. The support of theses features for specific classes would now be combined to provide an overall indicator of support for the class.

C. PROBLEMS WITH THE KNOWLEDGE BASED SYSTEM APPROACH

The use of a rule base and a priori probabilities for feature support of classes can be considered an expert system approach to the situation assessment task. The KBS conducts the actual analysis, the analyst merely queries the system. The problem with an expert-type system approach is similar to that of a Bayesian approach, an endless task of defining and updating the rule base.

Also, the rules used to determine the support of a feature for a class are not well defined. Table 5.4 showed hypothetical values of feature support for the attack intention. The adjectives used, "massing" and "extensive", are very indistinct. The rules using them would then require some way of inferencing this indistinct adjective from the data base ("fuzzy sets"). For example, extensive preparation would need to be defined as number of artillery rounds in an hour, or number of targets engaged in a specific time period. If the data base supported these criteria, then the rules base could deduce that extensive preparation has occurred.

Unfortunately, this use of explicit rules and indistinct features in an expert system can lead to a false sense of security of the battlefield. The analyst is dependent on the internal design of the system. He is receiving the system's analysis and may tend to discredit his own interpretation of the situation. Further, it is doubtful whether

any commander is presently ready to allow his analyst to accept a situation assessment from a "black box". The commander has too much risk associated with his decision based on this analysis to accept a fully automated analysis. Cushman, [Ref. 1], suggests that a commander will not be inclined to accept black box analysis , but instead support gradual automation based on transparent aids which allow the analyst to retain control.

A decision aid, similar to that in Chapter IV, used in a quick assessment capability, is a complimentary alternative to the Knowledge Based System. This aid would act more as a parallel process with the analyst reflecting his view of the battlefield rather than an "expert" view given by the rules created by the "experts". The analyst is the expert in his division and cannot rely on rules or features created by other.

Furthermore, the features suggested by Ben-Basset, [Ref. 17: p.486], were not intended for dogmatic application in all battlefield situations, but are given as a guide to the analyst for the analyst to use [Ref. 18: p. T-1]. The Dempster-Shafer aid would be dependent on the analyst and his inputs rather than the analyst depending on the expert system. While the analyst would use these indicators as guides to assign Mass values, he would be free to change support values based on his knowledge of the situation and his prior experience.

D. THE DEMPSTER-SHAFER THEORY IN SITUATION DEVELOPMENT ANALYSIS

The Dempster-Shafer theory could be used in two ways in the DSDA. First, it could be used within the Knowledge Based Structure proposed by Baltezore. Dempster-Shafer would be the combination technique used to aggregate the

feature probabilities and produce the overall Belief for each class. The feature probabilities would have to be represented as Masses and adhere to the definition of Mass presented in Chapter III. The use of the Gordon and Shortliffe method was discussed in Chapter IV using the example of determine overall enemy intentions. This technique would reduce internal Knowledge Based System inference time and provide an efficient combination of evidence.

Second, the aid could be used as a quick assessment device for the analyst. This use would be most relevant when the analyst can narrow the scope of the class. If the general class was attack and the analyst was concerned with the sector of attack, the Dempster-Shafer aid would be used to determine Belief for sector possibilities. This scope of use was represented through examples in Chapters III and IV using the sector of attack example.

The aid would serve as a reflection of the analyst's assessment of the battlefield as time progresses and reports are received. Using Belief values, the analyst will recognize the most likely sector of attack and advise the commander. Using Ignorance values, the analyst will reposition or reorient sensors to investigate lack of knowledge of activity.

1. Advantages Of The Quick Assessment Use

There are three main advantages to the quick assessment use over the knowledge based system:

a) Absence of Rule Base
b) Absence of Data Base
c) Speed

a. Absence of Rule Base

The analyst is not dependent on a predefined rule base to deduce support for the sector of attack. He

will consider each report and assign Mass values based on his understanding of the enemy. This understanding will come from battlefield experience and knowledge gained from experts through instruction.

b. Absence Of Data Base

The reports received in the quick assessment capability will still be stored in some data base for future use. However, the analyst would not search through previous data, but interpret each piece as it is received.

c. Speed

For each rule used, the KBS must satisfy the precedent (if portion), to allow use of the antecedent (then portion). This process requires the continual search of the data base for conditions that satisfy the rule precedent. With a large rule base, such as that needed in the KBS structure to define all types of enemy activity, the cycling time would be prohibitive for a real time assessment capability.

If time is available, the KBS procedure would be used to determine enemy intent, but if a quick assessment is desired, as discussed in Chapter II, the alternative capability of Dempster-Shafer is the better option.

E. RELEVANCY OF DEMPSTER-SHAFER TO SITUATION DEVELOPMENT ANALYSIS

It should be obvious that the examples used in preceding chapters are the same as the mission objectives of the Situation Development Analyst. The use of the Dempster-Shafer theory in this specific part of the Intelligence node will speed the analytical process, providing the commander fast and accurate intelligence

support. This enhancement will reduce command and control cycle time and gain the benefit of this reduction discussed in Chapter II.

VI. SUMMARY AND CONCLUSIONS

A. SUMMARY

This thesis has demonstrated the use of the Dempster-Shafer theory of evidence in a decision aid to reduce command and control cycle time on the battlefield. The reduced cycle time allows the battlefield commander to interdict the enemy force earlier and gain a decisive advantage.

The command and control cycle was modelled as a network to investigate data flow and network processes. The Interpretation node was determined to be the critical node and also a candidate for enhancement through application of the Dempster-Shafer theory. The Dempster-Shafer approach was presented as a plausible evidence combination technique when uncertain, incomplete, and incorrect evidence must be combined in a battlefield environment.

Three methods for reducing the computational complexity, of the Dempster-Shafer theory, Barnett, Gordon and Shortliffe, and multiple Frames of Discernment, were demonstrated. These methods all have restrictions involving trade-offs between flexibility or scope and time efficiency. Military examples that met these restrictions were presented to demonstrate their possible use on the battlefield. A decision aid based on the Dempster-Shafer theory was created and discussed. The aid eased the computational burden of Dempster-Shafer and allowed comparisons of computational speed with the three reduction methods.

A specific application area for the Dempster-Shafer theory, Situation Assessment in the All Source Analysis System (ASAS) was described. The task of situation

assessment provided a good example of a process in the Interpretation node that can be enhanced by Dempster-Shafer.

B. CONCLUSIONS

Dempster-Shafer was not presented as a "cure-all" for evidence combination. Nor was it presented as a replacement for more probabilisticly rigorous techniques such as, Bayesian Inference. The use of Dempster-Shafer in this thesis showed a logical method of combining data on the battlefield to help the analyst determine enemy intentions. The flexibility of Dempster-Shafer in handling battlefield-quality evidence should not be lost in discussions over its shortcomings. The intelligence analyst and battlefield commander need to make the best use of all data available in the most efficient and accurate manner possible. Dempster-Shafer is a viable technique to assist in this process.

The use of Dempster-Shafer in the Knowledge Based System proposed by Baltazore would allow the intelligence analyst to conduct automated analysis. The analyst would access the data base of evidence and receive Belief values for his hypotheses of enemy intentions. The use of the Barnett, Gordon and Shortliffe, or multiple Frames of Discernment methods, when applicable, would allow for the most rapid computation of Belief.

Automated analysis in this manner would allow access to more data than the human processor could handle. Many analysts could use the same system over a period of time to analyze trends in the enemy activity. Furthermore, the analyst now has a backup system to his manual method of analysis. He can make the most use of the human-machine leverage discussed in Chapter II.

C. AREAS FOR FURTHER ANALYSIS

This thesis has only begun to investigate the use of Dempster-Shafer on the battlefield. The determination of mass values based on receipt of evidence has been left to the analyst. The task of determining these values for Dempster-Shafer, or any combination method, is formidable. Chapter V discussed the problem of a rule base with explicitly assigned probabilities. Unfortunately there does not exist a data base from which probabilities of enemy intentions based on tactics can be extracted.

Samet, [Ref. 19], has said that each sensor report has a reliability and accuracy associated with it. These features could weight the mass values assigned by the expert or analyst. The integration of these reliability and accuracy values into the Dempster-Shafer mass values has not been discussed.

The Knowledge Based System hardware of a Decision Support System, such as that proposed by Baltezore, must be designed to accept an evidence combination technique.

These are but a few of the areas that are left for further exploration of Dempster-Shafer on the battlefield.

APPENDIX A

THE DEMPSTER-SHAFER DECISION AID

This appendix contains the listing of the code created for the decision aid and an explanation of the procedures. In general, the aid is a continuous loop that prompts the user for input from each knowledge source report received. It requests set identification and mass assigned to that particular set. The aid then combines the new input with the current mass values using Dempster-Shafer. New Beliefs are computed only if directed by the user and then displayed to reveal the current status of knowledge about the Frame.

The program is written in Turbo PASCAL for an IBM compatible computer. A main user routine and fourteen procedures, called throughout the routine, make up the program. See figure A.1 for the program's Flow Diagram.

As stated, the program is composed of the following procedures.

A. MAIN PROGRAM

The main program, DS(Dempster-Shafer), is an executive-like program that reacts to the user's desires. It initially sets up the PASCAL Record Structure that will contain those items of information necessary for use in the program by Dempster-Shafer computations. These items are:

1. Mass (Current Mass).
2. Tempmass (Temporary storage for Combined Mass).
3. Newmass (User Mass input from Knowledge Source).
4. Belief (Current Belief).
5. ID (Set Describer).
6. Value (Number of Items in Set ID).

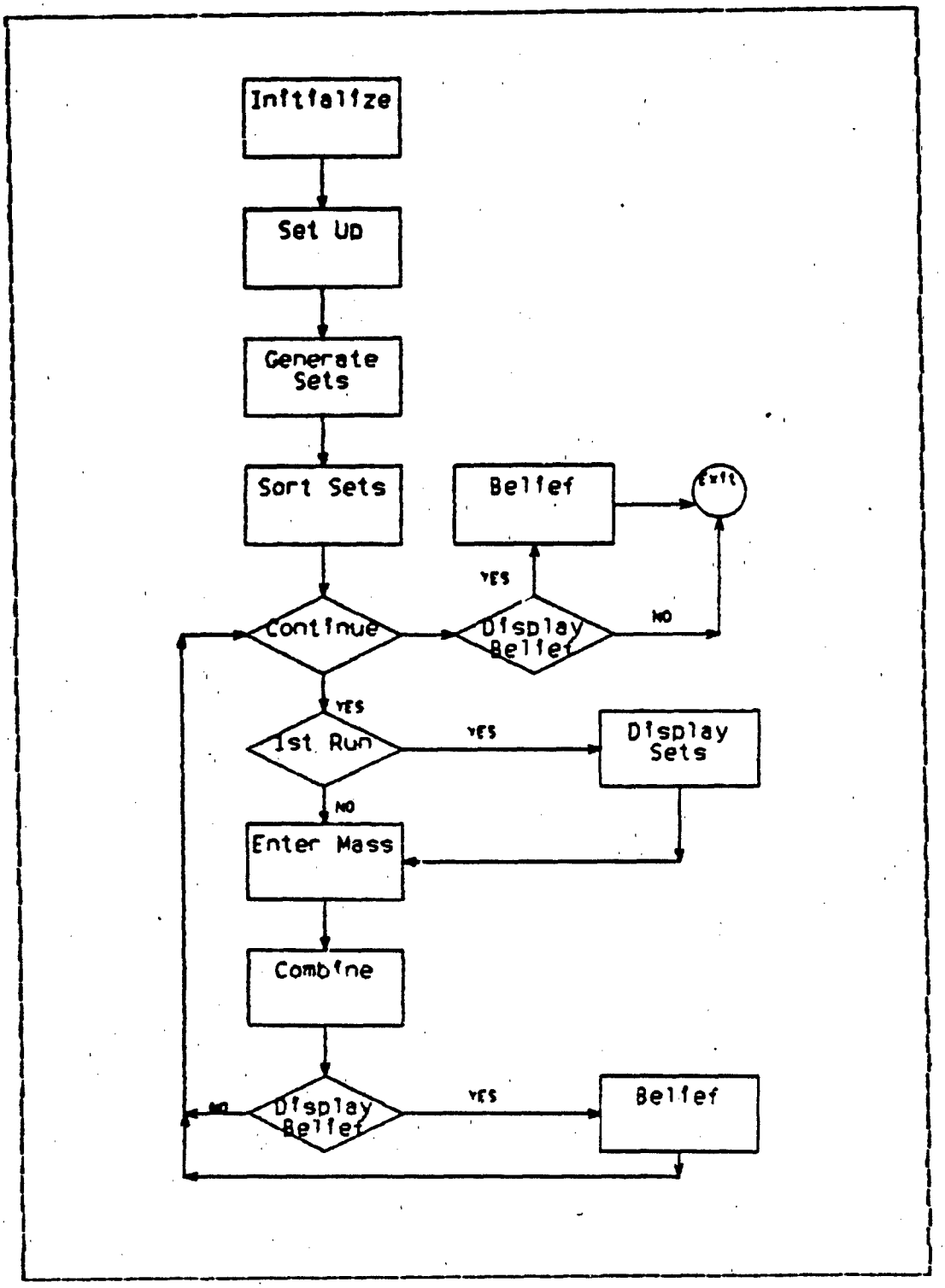

Figure A.1 Flow Diagram for the Decision Aid Program.

An array is then created to allow for the maximum number of items in the Frame of Discernment which the computer can store.

Next, the main program begins a calling sequence of the procedures. Within one of these procedures, many other procedures may be called to accomplish the task at hand. After the aid is ready for user input, the main program asks if the user desires to continue. If the answer is yes, then the program cycles again. If no, then the user is allowed to request a final display of the output (Belief, Plausibility, and Mass) and the program terminates.

The first procedure called by the program is Copyrite.

B. PROCEDURE COPYRITE

Procedure Copyrite is just a "cover sheet" for the aid identifying the theory used in the aid, and the creator.

The next call is to procedure Initialize.

C. PROCEDURE INITIALIZE

Procedure Initialize requests the number of items in the user's Frame of Discernment. It then determines the letter of the alphabet corresponding to the end of the size of the Frame,i.e., 5 = E, 7 = G. The letters of the alphabet will then be used as identifiers for each separate set. This use of letters allows for set operations within PASCAL.

The number of sets that can be made from the Frame is then computed, where size = 2 exp N, N = number of items in the frame.

Procedure Set Up is called next.

D. PROCEDURE SET UP

This procedure sets up the first two subsets of the Frame of Discernment. Those sets are the entire Frame, letters "A" through the end letter, and the null set, letter "Z", used only for easy identification.

Procedure Levels is next.

E. PROCEDURE LEVELS

Procedure Levels is used to compute the number of combinations of set size N split into subsets of size N through 1. In other words, N items N at a time, N items N-1 at a time, through N items 1 at a time.

The results of this procedure are stored in an array and are used to help reduce "do-loop" cycle time. In a loop where it is necessary to check subsets of the current set, as when Belief is computed, the size of the current set is determined by its Value (number of items in set). Then only the sets with value less than this are checked for subset possibility. This action eliminates unnecessary checks of parent sets that cannot be subsets of the current set.

The next call is to Procedure Generate.

F. PROCEDURE GENERATE

Procedure Generate is a complex recursive routine to create all subsets of the Frame of Discernment including the null set. When the Frame is large, it is the most time consuming procedure in the program. Figure A.2 shows a plot of some procedure execution times versus number of items in the Frame of Discernment.

Starting with the set of the entire Frame, Procedure Generate creates new sets by removing one character at a time from the current set. All subsets of the new set are then created by a recursive call to generate.

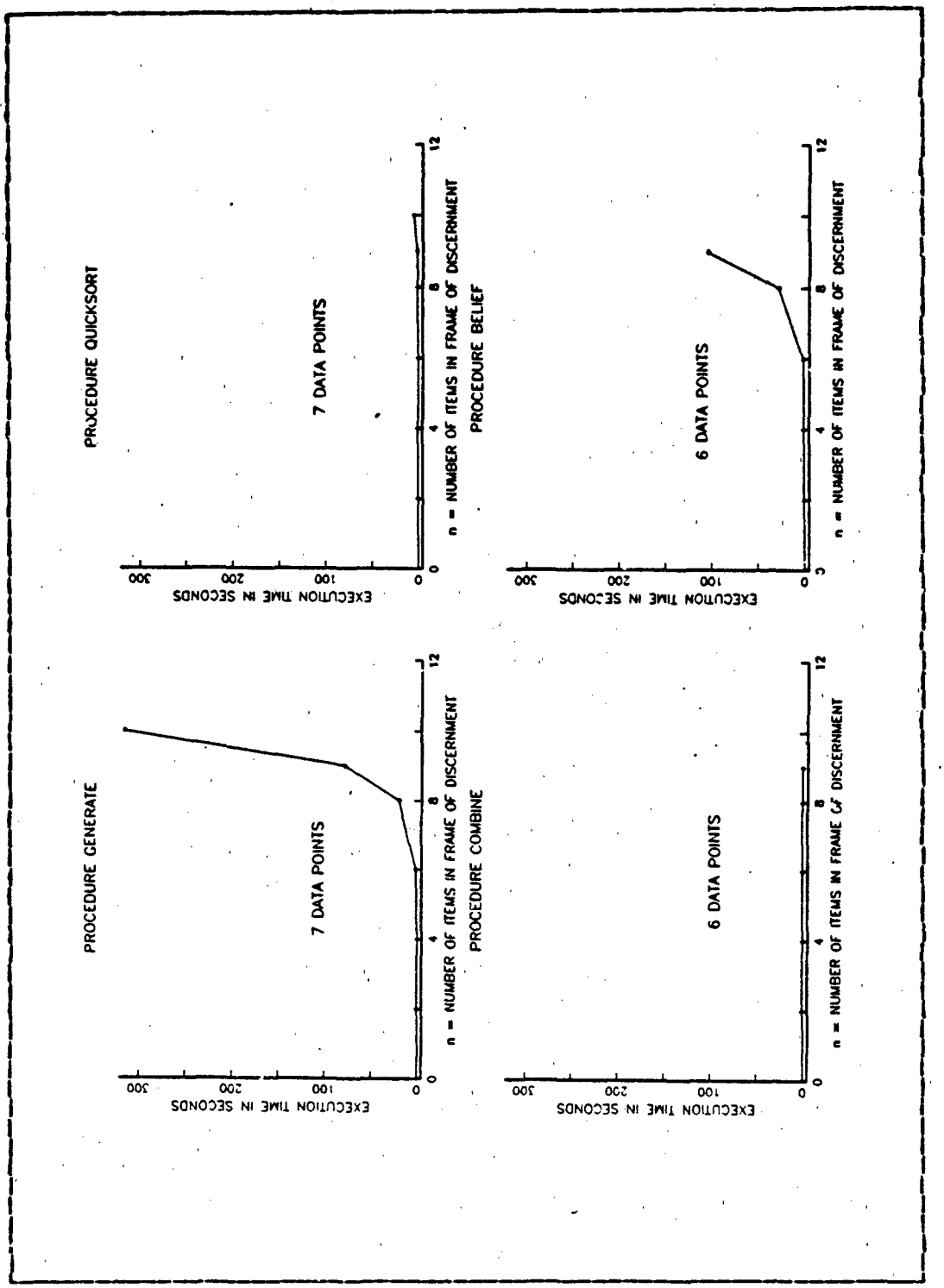

Figure A.2 Execution Times Versus Frame Size.

The computation time is high since identical sets are created in the recursive calls. Duplicate sets must be checked for and eliminated. As the frame size grows, this time factor is compounded, see Figure A.2.

The next procedure called is Quicksort.

G. PROCEDURE QUICKSORT

Quicksort, like Generate, is a recursive routine that sorts the sets according to the number of items in the set ID. This resulting sorted array of records allows for a more efficient check for subsets. A subset can only occur at a lower level of Value (number of items in set ID) than its superset. Therefore, loops computing Belief or Plausibility need only check lower levels of sets and not the entire tree of sets.

Quicksort is almost unaffected by the number of items in the Frame of Discernment due to its efficiency, see Figure A.2.

If the user desires to continue with the program, Procedure Entermass is then called.

H. PROCEDURE ENTERMASS

Entermass is the main user input procedure that assigns new mass values to those sets identified by the current knowledge source.

The first time called, Entermass allows the user to request a display of all the various sets that can occur. The user may then print this list for future reference.

The aid then prompts the user for the identity of the set and the mass to be assigned to that set. Set identities are checked to insure they are in the domain of the Frame of Discernment. Mass is checked for containment in the set [0,1].

Total mass entered during the current session is checked to make sure it does not exceed 1.0. Set ID are also checked to see if mass has already been assigned to the set during this session. If so, the user can change the mass value, or leave it at the current value.

When the total mass entered equals 1.0, the new masses are combined with the old mass value.

This combination is done using Procedure Combine.

I. PROCEDURE COMBINE

This procedure combines masses using Dempster's Rule of Combination and then normalizes the new masses. Current masses(masses at the end of the previous session) are then replaced with the new combined mass values.

Time of execution of the procedure is reduced by limiting the combination to only those sets that have a mass from the previous session, and those sets that were assigned masses during the current session. This efficient operation keeps execution time to a minimum, see Figure A.2.

The intersection-set of sets with masses are then identified and assigned the new masses. After all orthogonal sums (See Chapter III) are computed, new masses are assigned back to the Record for each set.

If there is mass assigned to the null set, by default the mass assigned to the other sets does not total 1.0, then the masses are normalized.

The Normalizing Factor is displayed to the user. New masses are calculated and assigned to the sets. The user is then asked if computation of Beliefs is desired.

J. PROCEDURE BELIEF

The computation of Belief is a very time consuming process since the mass for all subsets of each set must be

summed. The execution time for Procedure Belief is shown in Figure A.2.

Both Value (part of the Record) and Levels (created by a procedure) are used to reduce the search time for subsets to only those sets with the possibility of being a subset. Only those sets at lower levels of the Record structure may be subsets. After Belief values are computed, they are assigned back to the Record for each set.

The Belief, Plausibility, and Mass for each set are then displayed to the user.

K. PROCEDURE DISPLAY

Display is a procedure that writes all sets, from higher order to lower, to the screen with their respective Belief, Plausibility, and Mass. However, only those sets with Belief values greater than zero are displayed. The screen will display only 12 sets at a time to allow easy viewing and, if desired, printing by the user.

After this step, the masses from the next knowledge source are then entered and the process continues.

L. ADDITIONAL PROCEDURES

There are several procedures used throughout the program not mentioned above that are described here.

1. Procedure Display2

Display2 is the procedure used in the first iteration of Entermass that, if requested by the user, displays all the sets (only set IDs).

2. Procedure Checkanswer

Checkanswer is a procedure used to insure the answer to Yes or No questions is in the correct form.

3. Procedure Maxinteger

Maxinteger is used in the Combine Procedure to limit the search for the set containing the result of the intersection of two masses. Since the sets are sorted, it is only necessary to look in the lowest level(or highest value in a high to low sorted array) of the two set IDs combined.

4. Procedure Factorial

This procedure is used in Procedure Levels and computes the factorial of a number. This result is used in the computation of combinations for the values in the Levels array.

Listing of the PASCAL program used in the Decision Aid.

```
program ds;

const
     tempsize = 512;

type
    node = record     {stores data relative to d-s}
         mass : real;
         tempmass : real;
         newmass : real;
         belief : real;
         plausible : real;
         id : set of char;
         value: integer;
         end;{node}

    branch = array[1..tempsize] of node;

var
   count,oldcount,itemnumber,numitems,ii,size,cntr, one :
integer;
   k,enditem : char;
   tree : branch;
   goodanswer, continue : boolean;
   ch44,contanswer, beliefanswer : char;
   levelend : array[1..20] of integer;
   tempmasscntr : array[1..25] of integer;
   oldmasscntr : array[1..tempsize] of integer;


{include the following procedures in this main program}

{$i belief.pas}
{$i factorial.pas}
{$i levels.pas}
{$i copyrite.pas}
{$i generate.pas}
{$i setup.pas}
{$i checkanswer.pas}
{$i initialize.pas}
{$i quicksort.pas}
{$i sort.pas}
{$i display.pas}
{$i display2.pas}
{$i maxinteger.pas}
{$i combine.pas}
```

```
{$i entermass.pas}


begin {main}

itemnumber:=0; {keeps track of the number of evidence
combined}

{begin calling sequence}

copyrite;
initialize;
setup;
levels(numitems);

ch44:= 'a';

writeln;
textcolor(14);
writeln('   Generating Subsets .............');
generate(tree[2],ch44);
writeln;
writeln('   Sorting Subsets ................');
textcolor(15);

one:= 1;

quicksort(one,size);

for ii:= 1 to size do {initialize set values}
    begin
    tree[ii].belief:= 0.0;
    tree[ii].mass:= 0.0;
    end;{for}

clrscr;
writeln;
writeln('**************************************************
**********');
writeln('Your Frame of Discernment has been expanded into
all Subsets');
writeln('and the Null Set');
writeln('As Each Item Of Evidence Is Received, You Will Be
Prompted ');
writeln('For The Mass Distribution');
writeln('**************************************************
**********');
writeln;

{check for users desire to continue}
```

```
continue:= true;
while continue do
      begin

      repeat {until goodanswer}
         writeln;
         writeln('====> Do You Wish To Continue? Y or N');
         textcolor(12);
         writeln('        A No Answer Will Exit Program');
         textcolor(15);
         read (kbd,contanswer);
         writeln (contanswer);

         checkanswer(contanswer,goodanswer);{checks to see
if in Y,y,N,n}

      until goodanswer;

      if contanswer = 'n' then
         continue:= false;

      if continue then
         entermass(itemnumber,numitems);
      writeln;

      repeat {until goodanswer}
         writeln;
         writeln;
         writeln ('====> Do You Wish To Display Belief? Y or
N ');
         read (kbd,beliefanswer);
         writeln (beliefanswer);
         checkanswer(beliefanswer,goodanswer);

      until goodanswer;

      if beliefanswer = 'y' then
         begin
         belief;
         display;
         end;{if}

end;{while}

end.{main}

procedure setup; {creates initial 2 sets, null and
theta(entir frame)}

begin
tree[1].id := ['z']; {null set}
```

```
tree[1].value:= Ø;
tree[2].id := ['a'..enditem];  {entire frame}
tree[2].value:= numitems; {number of elements in set id}
cntr:= 2;

end;{setup}
procedure initialize; {user procedure to determine size of
frame}

var
   i,j : integer;
   ok : boolean;

begin
for j:= 1 to 5 do
    writeln;

repeat {until ok}
  writeln('====> Enter Number of Items in Frame of
Discernment');

  {$I-} readln (numitems) {$I+};
  ok:= (IOresult = Ø);
   if not OK then
     begin
     textcolor(12);
     writeln('*** Improper answer, retry');
     textcolor(15);
     end;{if}

until ok;

writeln (numitems);
enditem:= chr(numitems + ord('a')-1);
writeln;
writeln('*** Frame of Discernment has ',numitems :3, ' items
and will be listed');
writeln;
writeln('as Character Set a through ',enditem );
size:= 2;
 for i:= 1 to numitems-1 do   {computes 2**n}
         size:= size * 2;

end;{initialize}
procedure copyrite; {cover display}

var
   ll : integer;
   dummy : char;
```

```
begin
     textcolor(15);
     textbackground(1);
     clrscr;
     for ll:=1 to 5 do
         writeln;
     writeln('          YOU ARE ABOUT TO USE A DECISION AID
CREATED TO COMBINE          ');
     writeln;
     writeln('          EVIDENCE USING THE DEMPSTER-SHAFER
THEORY OF EVIDENCE          ');
     writeln;
     writeln('                                   COMBINATION
');
     writeln;
     writeln;
     textcolor(12);
     writeln('  CAUTION!!: Only those willing to venture
beyond');
     writeln('             Bayesian Inference should
continue');
     textcolor(14);
     writeln;
     writeln;
     writeln('  Written by CPT William H. Cleckner, US Army
as a prototype');
     writeln('  decision aid for combining intelligence and
determining an');
     writeln('  enemy commanders main attack sector.');
     for ll:= 1 to 5 do
         writeln;
     textcolor(15);
     writeln('  Push any key to continue');
     read (kbd,dummy);
     clrscr;

end;
procedure generate(var rl : node; var beginitem : char);

{conducts a recursive call to generate subsets}


var
   i,j : integer;
   ch1,ch11 : char;
   newset : boolean;

begin

     for ch1:= beginitem to enditem do {remove a character
at a time until nullset reached}
```

```
          begin
          if ch1 in r1.id then
             begin
             tree[cntr + 1].id:= r1.id - [ch1];
             if tree[cntr + 1].id <> [] then
                begin
                cntr:= cntr + 1;
                newset:= true;

                for j := 1 to cntr - 1 do {check for repeat
character sets}
                   begin
                   if (tree[j].id = tree[cntr].id) and newset
then
                   begin
                   newset:= false;
                  cntr:= cntr - 1;
                   end;{if}
             end;{for}

          if (newset) then  {if set not created before,
generate its subsets}
              begin
              tree[cntr].value:= r1.value - 1;
              ch11:= succ(ch1);
              generate(tree[cntr], ch11);
              end;{if}

          end;{if}
       end;{if}
    end;{for}
end;{generate}
procedure quicksort(var first : integer; var last :
integer);

var
   i, j, dividingline : integer;
   temporary : node;

begin
     i:= first;
     j:= last;
     dividingline:= tree[(first + last) div 2].value;
     repeat
           while tree[i].value > dividingline do
                 i:= i+1;
           while tree[j].value < dividingline do
                 j:= j-1;
           if i <= j then
              begin
```

```
                    temporary:= tree[i];
                    tree[i]:= tree[j];
                    tree[j]:= temporary;
                    i:= i+1;
                    j:= j-1;
                end;

        until i > j;
        if first < j then quicksort(first, j);
        if i < last then quicksort(i, last);

end;{quicksort}

procedure entermass(var item : integer; var numitems :
integer);

{enter new evidence temporary sets}

label
     10,20;

var
   goodentry,goodanswer, ok : boolean;
   displayanswer,ch3,ch33,changeanswer : char;
   check,tmass : real;
   qq,nn,kk,jj,mm,i : integer;
   tempset : set of char;
   tempid : string[20];


begin

item:= item + 1;
clrscr;
writeln ('*** You are entering the mass distribution for
item number:');
writeln;
writeln ('--> ',item :4);
writeln;

        if item = 1 then  {ask user if needs display of
sets}
        begin

         repeat {until goodanswer}
           writeln ('====> Do you need a display or all the
sets? Y or N');
           writeln;
           writeln ('          This Feature Will Only Appear
Once so Use');
```

```
            writeln ('          PrtSc key for hard copy of
sets          ');
            writeln;
            read (kbd,displayanswer);
            writeln (displayanswer);

            checkanswer (displayanswer,goodanswer); {insure
YyNn}

        until goodanswer;

        clrscr;
            if (displayanswer = 'y') or (displayanswer =
'Y') then
                begin
                display2;  {display if requested}
                end;{if}
        end;{if}

for nn:= 1 to size-1 do
    begin
    tree[nn].tempmass:= 0.0;
    tree[nn].newmass:= 0.0;
    end;{for}

repeat
      goodentry:= true;
      check:= 0.0;
      count:= 1;

      while check < 0.9999999 do {end while is label 20}
          begin
          tmass:= 0.0;

          repeat {until goodanswer label 10}
             tempset:= [];
             writeln;

            repeat {until ok}
             writeln ('====> Enter Set ID : ');
             {$I-} readln (tempid) {$I+};
             ok:= (IOresult = 0);
               if not OK then
                  begin
                    textcolor(12);
                    writeln('*** Improper answer, retry');
                    textcolor(15);
                  end;{if}
            until ok;

             writeln (tempid);
```

```
                for ii:=1 to length(tempid) do  {check for
set in frame}
                    begin
                    if (tempid[ii] in ['a'..enditem]) and
not (tempid[ii] = ' ') then
                       goodanswer:= true
                    else
                       begin
                        goodanswer:= false;
                        textcolor(12);
                        writeln;
                        writeln ('*** Set ID not in Frame,
Retry');
                        textcolor(15);
                        goto 10;
                        end;{if}

                    tempset:= tempset + [tempid[ii]];
{create tempset of id}
                    end;{for}

{check for attempt to enter mass for set already accessed}

          for qq:= 1 to size-1 do
              begin
              if (tempset = tree[qq].id) and
(tree[qq].tempmass>0.0e+00) then
                 begin
                 repeat {until goodanswer}
                    writeln;
                    textcolor(12);
                    writeln('   You Have Already Entered A
Mass For Set ',tempid,' = ',tree[qq].tempmass :6:3);
                    writeln('   Do You Wish To Change This
Input? Y or N');
                    textcolor(15);
                    read(kbd,changeanswer);
                    checkanswer(changeanswer,goodanswer);
                 until goodanswer;
                    if (changeanswer = 'y') or
(changeanswer = 'Y') then
                       begin
                       check:=check - tree[qq].tempmass;
                       goto 10;
                       end
                    else
                       goto 20;

                 end;{if}
          end;{for}
```

```
10:     until goodanswer;


        repeat  {until ok}

           writeln;
           writeln ('====> Enter Mass for Set');
           {$I-} readln (tmass) {$I+};

           ok:= (IOresult = 0);
                if not OK then
                  begin
                  textcolor(12);
                  writeln('*** Improper answer, retry');
                  textcolor(15);
                  end;{if}

        until ok;

        writeln (tmass :8:5);

        for jj:= 1 to cntr-1 do   {assign new masses to
tree}
            begin
            if tree[jj].id = tempset then
              begin
              tree[jj].tempmass:= tmass;
              tempmasscntr[count]:= jj;
              end;{if}
            end;{for}
        writeln;
        count:= count + 1;

        check:= check + tmass;
        textcolor(14);
        write ('Total Mass = ',check :8:5);
        write ('  (*** Reminder: Total Mass Must B = 1.0
to exit loop ***)');
        textcolor(15);
        writeln;

  20 : end;{while}

     if (check <= 0.9999999) or (check >= 1.0000001) then
        begin
        goodentry:= false;
        writeln;
        textcolor(12);
        writeln ('*** Warning, Mass Total =  ',check :4:2,'
Is Greater Than  1.0    ***');
```

```
        writeln;
        writeln ('*** You Must Input All Masses For This
Item Of Evidence ***');
        textcolor(15);
        writeln;

           for i:= 1 to count do
               tree[tempmasscntr[i]].tempmass:= 0.0;

        end;{if}

     until goodentry;

if item > 1 then
   combine; {return newmasses in tree}

if item = 1 then      {do not combine}
   begin
   oldcount:= count;
   for kk:= 1 to count do
       begin
       tree[tempmasscntr[kk]].mass:=
tree[tempmasscntr[kk]].tempmass; {reassign newmasses to
tree}
       oldmasscntr[kk]:= tempmasscntr[kk];
       end;{for}
   end;{if}

end;{entermass}
procedure combine;

{combines evidence using dempster-shafer}

var
   h,hn,j,jj,i,ii,k,kk,j2,high,tempcount : integer;
   normalfactor, totalmass, tempmass : real;
   intersection : set of char;
   tempoldmasscntr : array[1..tempsize] of integer;

begin

textcolor(14);
writeln('   Combining Masses .....');
textcolor(15);

tempmass:= 0.0;
totalmass:= 0.0;
tempcount:= 0;

{ check for intersection and increment newmass by mass
product}
```

```
for j:= 1 to oldcount do
    begin
     for i:= 1 to count do
       begin
        intersection:= tree[oldmasscntr[j]].id *
tree[tempmasscntr[i]].id;
         if intersection <> [] then
            begin

maxinteger(tempmasscntr[i],oldmasscntr[j],high);
                for jj:= levelend[tree[high].value+1]+1 to
levelend[tree[high].value] do
                 begin
                   if tree[jj].id = intersection then
                      begin
                      tree[jj].newmass:= tree[jj].newmass +
tree[oldmasscntr[j]].mass * tree[tempmasscntr[i]].tempmass;
                      tempcount:= tempcount + 1;
                      tempoldmasscntr[tempcount]:= jj;
                         for ii:= 1 to tempcount-1 do
                             if tempoldmasscntr[ii] = jj then
                                tempcount:= tempcount-1;
                      end;{if}
                 end;{for}
            end;{if}
       end;{for}
    end;{for}

oldcount:= tempcount;
for kk:= 1 to oldcount do
    oldmasscntr[kk]:= tempoldmasscntr[kk];

for j2:= 1 to size-1 do
    tree[j2].mass := 0.0;

for n:= 1 to oldcount do {assign combined masses back to
tree}
    begin
    totalmass:= totalmass + tree[oldmasscntr[n]].newmass;
    tree[oldmasscntr[h]].mass:=
tree[oldmasscntr[h]].newmass;
    end;{for}

{if mass assigned to nullset (by default that totalmass
assigned to }
{sets <> 1.0) then normalize}

if totalmass < 1.0 then
   begin
      normalfactor:= totalmass;
```

```
      textcolor(14);
      writeln('Normalizing Factor = ',normalfactor :6:3);
      textcolor(15);

      for k:= 1 to oldcount do
        tree[oldmasscntr[k]].mass:=
tree[oldmasscntr[k]].mass/normalfactor;
   end{if}

else
    writeln('   Normalizing Factor = ',totalmass :6:3);

end;{combine}
procedure belief; {sums masses of all subsets and assigns to
parent}

var
   i,i2,n,n2,notvalue : integer;
   d : char;
   notset : set of char;

begin
textcolor(14);
writeln('   Computing Beliefs.....');
textcolor(15);

for n:= 1 to size-1 do
    begin
    tree[n].belief:= tree[n].mass;
    for n2:= ((levelend[tree[n].value])+1) to size-1 do
{only checks its mass(n) and those sets with}
        begin
        if tree[n2].id <= tree[n].id then
           tree[n].belief:= tree[n].belief + tree[n2].mass;
{if subset then increment belief}
        end;{for}
    end;{for}

for i:= 2 to size-1 do
    begin
    notset:= ['a'..enditem] - tree[i].id;
    notvalue:= abs(tree[i].value - numitems);

       for i2:= (levelend[notvalue + 1] + 1) to
levelend[notvalue] do
           begin
           if tree[i2].id = notset then
              tree[i].plausible:= 1 - tree[i2].belief;
           end;{for}
    end;{for}
```

```
end;{belief}
procedure display; {displays set ids, beliefs, masses}

var
   jj : integer;
   ch,dummy : char;

begin
     clrscr;
     for jj:= 1 to size-1 do
       if tree[jj].belief > 0.0 then
         begin
         textcolor(13);
         writeln('----------------------------------------
---------------------------------');
         textcolor(15);
         write ('  Set ID: ');
         textcolor(14);
         for ch:= 'a' to enditem do
                begin
                if ch in tree[jj].id then
                   write(ch);
                end;{for}
         textcolor(15);
         write ('  Belief =
');textcolor(14);write(tree[jj].belief :6:3);
         textcolor(15);
         write ('  Plausibility =
');textcolor(14);write(tree[jj].plausible :6:3);
         textcolor(15);
         write ('  Mass =
');textcolor(14);write(tree[jj].mass :6:3);
         textcolor(15);
         writeln;

         if (jj mod 12) = 0 then  {allows scrolling to stop
temporarily}
            begin
            textbackground(14);
            write('Push Any Key to Continue Display');
            textbackground(1);
            writeln;
            read(kbd,dummy);
            end;{if}
         end;{for}
end;{display}
procedure display2; {displays only set ids to choose from in
procedure entermass}

var
```

```
   jj : integer;
   ch : char;

begin
     clrscr;
     for jj:= 1 to size-1 do
         begin
         write ('  Set ID:  ');
         for ch:= 'a' to enditem do
                begin
                if ch in tree[jj].id then
                   write(ch);
                end;{for}
        writeln;
        end;{for}
end;{display2}

procedure levels(var itm : integer);

var
   tlevel : array[1..20] of integer;
   i : integer;
   numerator : real;

begin
levelend[itm + 1]:= 0;
numerator:= factorial(itm);
for i:= 1 to itm do
    tlevel[i]:= trunc(numerator / ( factorial(i-1) *
factorial(itm-i+1) ));

for i:= 1 to itm do
    levelend[itm-i+1]:= tlevel[i] + levelend[itm-i+2];
end;{levelend}



procedure checkanswer(var answer : char; var test :
boolean);

{checks to see if answer in set Y y N n returns false if
not}

begin
if answer in ['y','Y','n','N'] then
   test:= true
else
    begin
    test:= false;
    writeln('*** Answer was not in acceptable form, retry');
    end;{else}
```

```
end;{checkanswer}

procedure maxinteger(var a : integer; var b : integer; var
largest : integer);

{determines the maximum of two integers (used in procedure
combine)}

begin
if a > b then
   largest:= a

else
    largest:= b;

end;{mininteger}
function factorial (n : integer) : real;

var
   fact,j : real;
   i : integer;
begin

fact:= 1;

for i:= 1 to n do
    fact:= fact * i;

factorial:= fact;

end;
```

APPENDIX B

THE METHODS OF BARNETT AND GORDON AND SHORTLIFFE

This appendix contains the listing of the code created for the methods of Barnett and Gordon and Shortliffe described in Chapter IV. Note that Step 2 of Gordon and Shortliffe is not complete. It was written to compute Relief computation times.

Listing of the PASCAL program for Barnett's method.

```
program dsbarnett;

const
     tempsize = 100;

type
    node = record
         id : set of char;
         p : real;
         c : real;
         r : real;
         d : real;
         ptemp : real;
         ctemp : real;
         bel : real;
         belcomp : real;
         end;{node}

    branch = array[1..tempsize] of node;

var
   position,item, numitems, size, i, j, k : integer;
   dprod, cprod, normfac, s, sprime, ratio, bigc, bigd :
real;
   tree : branch;
   tempid : string[20];
   finished, done : boolean;
   donans, finans,n,y,enditem, dummy, ch : char;

begin

writeln('enter number of items in frame');
readln(numitems);
enditem:= chr(numitems + ord('a')-1);
size:= numitems;

for i:= 1 to size do
    begin
         with tree[i] do
             begin
             id:= [chr(i + ord('a')-1)];
             p:= 0.0;
             c:= 0.0;
             r:= 0.0;
             d:= 0.0;
             ptemp:= 0.0;
             ctemp:= 0.0;
```

```
                end;{with}
    end;{for}


done:= false;
repeat{until done}


finished:= false;
repeat{until finished}
writeln('enter set id');
readln(tempid);
   for j:= 1 to size do
       begin
       if [tempid] = tree[j].id then
            position:= j;
       end;{for}

 writeln('enter mass that supports: ',tempid);
 readln(tree[position].ptemp);
 writeln('enter mass that supports compliment of: ',tempid);
 readln(tree[position].ctemp);
 writeln;
 writeln('Step One and Step Two.........');

    with tree[position] do
        begin
        s:= 1-((1-ptemp) * (1-p));
        sprime:=1-((1-ctemp) * (1-c));
        p:=(s*(1-sprime))/(1-s*sprime);
        c:=(sprime*(1-s))/(1-s*sprime);
        r:=1-p-c;
        d:=c+r;
        writeln('p,c,r,d',p,c,r,d);
        end;{with}


writeln('finished? y or n');
readln(finans);
if (finans = 'y') or (finans = 'Y') then
   finished:= true;

until finished;
writeln;
writeln;
writeln('Step Three.................');

bigc:=1.0;
bigd:= 1.0;
ratio:=0.0;
```

```
for k:= 1 to size do
     begin
     with tree[k] do
          begin
          ratio:= ratio + p/d;
          bigd:= bigd * d;
          bigc:= bigc * c;
          end;{with}
     end;{for}

ratio:= 1 + ratio;
normfac:= 1/((bigd * ratio)-bigc);

writeln('ratio, normfac, bigd,  bigc,
',ratio,normfac,bigd,bigc);
for i:= 1 to size do
     begin
     cprod:= 1.0;
     dprod:= 1.0;
     for j:=1 to size do
         if j <> i then
            begin
            cprod:= cprod * tree[j].c;
            dprod:= dprod * tree[j].d ;
            end;{if}
     with tree[i] do
          begin
          bel:= normfac*((p*dprod) + (r*cprod));
          belcomp:= normfac * ( bigd * (ratio - 1 - (p/d)) +
c*dprod - bigc);
          end;{with}
     end;{for}
     readln(dummy);
      clrscr;
      for j:= 1 to size do
          begin
          textcolor(13);
          writeln('-----------------------------------------
---------------------------------');
          textcolor(15);
          write ('  Set ID: ');
          textcolor(14);
          for ch:= 'a' to enditem do
                 begin
                 if ch in tree[j].id then
                    write(ch);
                 end;{for}
          textcolor(15);
          write ('  Belief =
');textcolor(14);write(tree[j].bel :6:3);
          textcolor(15);
```

```
        write ('  Plausibility = ');textcolor(14);write(1-
tree[j].belcomp :6:3);
        writeln;

        if (j mod 12) = 0 then  {allows scrolling to stop
temporarily}
           begin
           textbackground(14);
           write('Push Any Key to Continue Display');
           textbackground(1);
           writeln;
           read(kbd,dummy);
           end;{if}
        end;{for}

writeln('done-end program, y or n?');
readln(donans);
if (donans = 'y') or (donans = 'Y') then
   done:= true;

until done;

end.{main}
```

Listing of the PASCAL program for Gordon and Shortliffe method.

```
program dsshort;

const
     tempsize = 100;

type
    node = record
         id : set of char;
         p : real;
         c : real;
         rp : real;
         rc : real;
         ptemp : real;
         ctemp : real;
         bel : real;
         end;{node}

    branch = array[1..tempsize] of node;

var
```

```
   tempset : set of char;
   position,item, numitems, size, i, j, k : integer;
   dprod, cprod, normfac, s, sprime, ratio, bigc, bigd :
real;
   tree : branch;
   setid, tempid : string[20];
   finished, done : boolean;
   donans, finans,n,y,enditem, dummy, ch : char;

begin

writeln('enter number of items in theta');
readln(numitems);
enditem:= chr(numitems + ord('a')-1);
writeln('enter number of sets to include theta');
readln(size);

for i:= 1 to size do
    begin
    with tree[i] do
         begin
         p:= 0.0;
         c:= 0.0;
         rp:= 0.0;
         rc:= 0.0;
         ptemp:= 0.0;
         ctemp:= 0.0;
         bel:= 0.0;
         end;{with}
    end;{for}

tree[1].id:= ['a'..enditem];
for i:= 2 to size do
    begin
    tempset:=[];
    writeln('enter set id for set number ',i);
    readln(setid);
    for j:= 1 to length(setid) do
        tempset:= tempset + [setid[j]];

         with tree[i] do
             begin
             id:= tempset;
             end;{with}
    end;{for}


done:= false;
repeat{until done}
```

```
finished:= false;
repeat{until finished}
writeln('enter set id');
readln(tempid);
     tempset:=[];
     for j:= 1 to length(tempid) do
                    tempset:= tempset + [tempid[j]];

   for j:= 2 to size do
       begin
       if tempset = tree[j].id then
            position:= j;
       end;{for}

writeln('enter mass that supports: ',tempid);
readln(tree[position].ptemp);
writeln('enter mass that supports compliment of: ',tempid);
readln(tree[position].ctemp);
writeln;
writeln('Step One.....');

   with tree[position] do
        begin
        s:= 1-((1-ptemp) * (1-p));
        sprime:=1-((1-ctemp) * (1-c));
        p:=(s*(1-sprime))/(1-s*sprime);
        c:=(sprime*(1-s))/(1-s*sprime);
        rp:=1-p;
        rc:=1-c;
        end;{with}


writeln('finished? y or n');
readln(finans);
if (finans = 'y') or (finans = 'Y') then
   finished:= true;

until finished;
writeln;
writeln('Step Two.....');

for i:=1 to size do
    begin
    tree[i].p:= tree[i].p * tree[i].rp;
    writeln('p,rp ',tree[i].p,tree[i].rp);
    end;{for}

for i:= 2 to size do
    begin
    for j:= 2 to size do
        begin
```

```
        if(j <> i) or not(tree[i].id<=tree[j].id) then
             tree[i].p:= tree[i].p * tree[j].rp;
        end;{for}
    end;{for}

normfac:= 0.0;
for i:= 1 to size do
    normfac:= normfac + tree[i].p;

writeln('normfac',normfac);
normfac:= 1/normfac;

for i:= 1 to size do
    tree[i].p:= tree[i].p * normfac;

writeln('Step 3.................');

for i:= 2 to size do
    begin
    for j:= 2 to size do
        begin
        if tree[j].id = ['a'..enditem] - tree[i].id then
           tree[j].p:= tree[j].p * tree[i].c;

        if tree[j].id <= tree[i].id then
           tree[j].p:= tree[j].p * tree[i].rc;
        end;{for}
    end;{for}

normfac:=0.0;
for i:= 1 to size do
    begin
    normfac := normfac + tree[i].p;
    end;{for}

for j:= 1 to size do
    begin
    tree[j].p:= tree[j].p/normfac;
    end;{for}

for i:= 1 to size do
    begin
    tree[i].bel:=0.0;
    for j:= 1 to size do
       begin
       if tree[j].id <= tree[i].id then
         tree[i].bel:= tree[i].bel + tree[j].p;
       end;{for}
    end;{for}
```

```
     clrscr;
     for j:= 1 to size do
          begin
          textcolor(13);
          writeln('----------------------------------------
--------------------------------');
          textcolor(15);
          write ('  Set ID: ');
          textcolor(14);
          for ch:= 'a' to enditem do
                begin
                if ch in tree[j].id then
                   write(ch);
                end;{for}
          textcolor(15);
          write ('  Belief =
');textcolor(14);write(tree[j].bel :6:3);
          textcolor(15);
          write ('  Mass = ');textcolor(14);write(tree[j].p
:6:3);
          writeln;

          if (j mod 12) = 0 then  {allows scrolling to stop
temporarily}
             begin
             textbackground(14);
             write('Push Any Key to Continue Display');
             textbackground(1);
             writeln;
             read(kbd,dummy);
             end;{if}
          end;{for}

writeln('done-end program, y or n?');
readln(donans);
if (donans = 'y') or (donans = 'Y') then
   done:= true;

until done;

end.{main}
```

LIST OF REFERENCES

1. Kirchhofer, Kirk H., "Military Command and Control, an Overview", Signal, August 1983, pp. 23-51.

2. TRADOC Operations Research Activity Decision Aids Product Review Board, by Jerome Jacobs, MAJ, USA, Team Leader, Fort Leavenworth, KS 1984.

3. Sweet, R., Frielander, Z., Hamilton, G. et al, The 1984 Measures of Effectiveness (MOE) for C^3I Evaluation Symposium: Proceedings and Analysis, MITRE Corporation, Bedford, MA, October 1984.

4. Stewart, S.R., G. Ross, and R. Tiede, Science Applications, Inc., Guidelines For Automating Command And Control In Field Units, ARI Field Office, Fort Leavenworth, KS, March 1984.

5. Miller, Richard S., A Framework for Establishing the Need for, and Value of C^2 Programs, a paper written at the Naval Postgraduate School and ammended at TRADOC Operations Research Activity, Monterey, CA, June 1980.

6. Lowrance, John D. and Thomas D. Garvey, Evidential Reasoning: A Developing Conflict, a paper written for Artificial Intelligence Center, SRI International, Menlo Park, CA.

7. Peterson, C.R., C.W. Kelly, T.H. Hazard, S. Barclay, Decision and Designs, Inc. Inference from Evidence: Bayes' Theorem, Defense Intelligence School, June 1973.

8. Land, Walker H., Jr. and David R. Woods, Critical Node Recognition For C^3CM Battle Systems Management, a paper submitted to the Military Operations Research Society (MORS), for the 53rd MORS Conference June 1985.

9. Dempster, Arthur P. "A Generalization of Bayesian Inference", Journal of the Royal Statistical Society, Series B, Vol. 30, pp 205-247, 1968.

10. Shafer, Glenn A., A Mathematical Theory of Evidence, Princeton University Press, Princeton, NJ 1976.

11. Dillard, Robin A., The Dempster-Shafer Theory Applied To Tactical Data Fusion In An Inference System, a paper prepared for Fifth MIT/ONR Workshop, Naval Ocean Systems Center, San Diego, CA 1982.

12. Barnett, Jeffrey A. Computational Methods For A Mathematical Theory Of Evidence, research conducted for the Defense Advanced Research Agency, USC/Information Sciences Institute.

13. Shafer, Glenn, and Amos Tversky, Languages and Designs for Probability Judgement, University of Kansas, Lawrence, KS and Stanford University, Stanford, CA 1985.

14. Garvey, Thomas D. Machine-Intelligence-Based Multisensor ESM System, SRI Project 1655, Menlo Park, CA December 1981.

15. Gordon, Jean and Edward Shortliffe, A Method For Managing Evidential Reasoning In A Hierarchical Hypothesis Space, a paper prepared for the Departments of Medicine and Computer Science, Stanford University, Stanford, CA., 21 September 1984.

16. Baltezore, Lawrence A., A Decision Support System Proposal For The Division's Situation Development Analyst, M.S thesis, Naval Postgraduate School, Monterey, California, December 1984.

17. Ben-Bassat, Moshe and Amos Freedy, "Knowledge Requirements and Management in Expert Decision Support Systems for (Military)Situation Assessment", IEEE Transactions On Systems, Man, And Cybernetics, Vol. SMC-12, No. 4, July/August 1982.

18. Department of the Army Field Manual 30-5, Combat Intelligence, October 1973.

19. Samet, Michael G., "Quantitative Interpretation of Two Qualitative Scales Used to Rate Military Intelligence", Human Factors, 17(2), 1975.

BIBLIOGRAPHY

Borland International, Turbo PASCAL User's Manual, Version 2.0, Scotts Valley, California.

Bonczek, Robert H., Clyde W. Holsapple, Andrew B. Whinston, Foundations of Decision Support Systems, Orlando: Academic Press, Inc., 1981.

Buchanan, Bruce G. and Edward H. Shortliffe, eds., Rule Based Expert Systems, Reading, Mass.: Addison-Wesley, 1984.

Dillard, Robin A., Tactical Inferencing With The Dempster-Shafer Theory Of Evidence, A paper prepared for Seventeenth Asilomar Conference on Circuits, Systems, and Computers, Naval Ocean Systems Center, San Diego, CA 1983.

Downes-Martin, S.G., Military Intelligence and Dempster-Shafer, A Working Paper prepared for the Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, 18 December 1984.

Eddy, William F. and Gabriel P. Pei, Structures Of Rule-Based Belief Functions, Department of Statistics, Carnegie-Mellon University, Pittsburgh, PA, 15 March 1985.

Garvey, Thomas D., Lowrance, John D., and Martin A. Fischler, An Inference Technique For Integrating Knowledge From Disparate Sources, SRI International, Menlo Park, CA.

Graham, Neill, Introduction to PASCAL, St. Paul: West Publishing Co., 1983.

Larson, Harold J. Introduction To Probability Theory And Statistical Inference, New York: John Wiley and Sons, 1982.

Shafer, Glenn, The Problem Of Dependent Evidence, A Working Paper prepared for the School Of Business, The University Of Kansas-Lawrence, October 1984.

Shafer, Glenn, Probability Judgement in Artificial Intelligence, A paper prepared for the School Of Business, The University Of Kansas-Lawrence, December 1984.

Sprague, R. H., Jr., and D. E. Carlson, Building Effective Decision Support Systems, Prentice-Hall, 1982.

INITIAL DISTRIBUTION LIST

No.		Copies
1.	Operations Research Department Attn: LCDR Paul S. Fischbeck, Code 30Fi Naval Postgraduate School Monterey, California 93943-5100	1
2.	Operations Research Department Attn: LTC Jack B. Gafford, Code 30Gf Naval Postgraduate School Monterey, California 93943-5100	1
3.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5100	2
4.	Captain William H. Cleckner, IV Department Of Mathematical Sciences U.S. Air Force Academy Colorado Springs, Colorado 83840	4
5.	Studies And Analysis Directorate Attn: Command and Control Analysis Division Fort Leavenworth, Kansas 66027	1
6.	Combined Arms Research Library U.S. Army Command and Staff College Fort Leavenworth, Kansas 66027	2
7.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
8.	Professor Edward Shortliffe Department of Computer Science Stanford University Stanford, California 94305	1
9.	Robin A. Dillard Naval Oceans Systems Center San Diego, California 92152	1
10.	LCDR Ted Kral Naval Oceans Systems Center San Diego, California 92152	1
11.	Professor Glen Shafer The School of Business University of Kansas-Lawrence Lawrence, Kansas 66048	1
12.	Joint C^3 Academic Group Attn: Professor M. P. Soveriegn, Code 55 Zo Naval Postgraduate School Monterey, California 93943-5100	1
13.	Roland Associates 500 Sloat Avenue Monterey, California 93940	1

14. Operations Research Department 1
Attn: Professor A. R. Washburn, Code 30 Ws
Naval Postgraduate School
Monterey, California 93943-5100