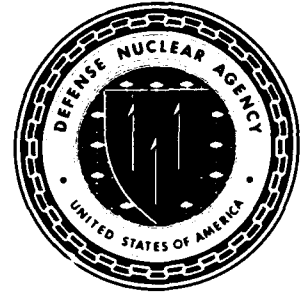


AD-A259 394



Defense Nuclear Agency
Alexandria, VA 22310-3398



DNA-TR-92-82

Artificial Intelligence Applications for Nuclear Survivability Validation

DTIC
ELECTE
JAN 21 1993
S C D

David M. Rodvold
Kaman Sciences Corporation
P.O. Box 7463
Colorado Springs, CO 80933-7463

November 1992

Technical Report

CONTRACT No. DNA 001-89-C-0080

Approved for public release;
distribution is unlimited.

93-01019



5381

98 ' 1 21 023

Destroy this report when it is no longer needed. Do not return to sender.

PLEASE NOTIFY THE DEFENSE NUCLEAR AGENCY,
ATTN: CSTI, 6801 TELEGRAPH ROAD, ALEXANDRIA, VA
22310-3398, IF YOUR ADDRESS IS INCORRECT, IF YOU
WISH IT DELETED FROM THE DISTRIBUTION LIST, OR
IF THE ADDRESSEE IS NO LONGER EMPLOYED BY YOUR
ORGANIZATION.



DISTRIBUTION LIST UPDATE

This mailer is provided to enable DNA to maintain current distribution lists for reports. (We would appreciate your providing the requested information.)

- ☐ Add the individual listed to your distribution list.
- ☐ Delete the cited organization/individual.
- ☐ Change of address.

NOTE:

Please return the mailing label from the document so that any additions, changes, corrections or deletions can be made easily.

NAME: _____

ORGANIZATION: _____

OLD ADDRESS**CURRENT ADDRESS**

TELEPHONE NUMBER: () _____

DNA PUBLICATION NUMBER/TITLE**CHANGES/DELETIONS/ADDITIONS, etc.)**
(Attach Sheet if more Space is Required)

DNA OR OTHER GOVERNMENT CONTRACT NUMBER: _____

CERTIFICATION OF NEED-TO-KNOW BY GOVERNMENT SPONSOR (if other than DNA):

SPONSORING ORGANIZATION: _____

CONTRACTING OFFICER OR REPRESENTATIVE: _____

SIGNATURE: _____

CUT HERE AND RETURN



DEFENSE NUCLEAR AGENCY
ATTN: TITL
6801 TELEGRAPH ROAD
ALEXANDRIA, VA 22310-3398

DEFENSE NUCLEAR AGENCY
ATTN: TITL
6801 TELEGRAPH ROAD
ALEXANDRIA, VA 22310-3398

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE 921101	3. REPORT TYPE AND DATES COVERED Technical 920101 - 920408		
4. TITLE AND SUBTITLE Artificial Intelligence Applications for Nuclear Survivability Validation		5. FUNDING NUMBERS C - DNA 001-89-C-0080 PE - 62715H PR - SB TA - SA WU - DH057780		
6. AUTHOR(S) David M. Rodvold				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Kaman Sciences Corporation P.O. Box 7463 Colorado Springs, CO 80933-7463		8. PERFORMING ORGANIZATION REPORT NUMBER K92-18U(R)		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Nuclear Agency 6801 Telegraph Road Alexandria, VA 22310-3398 OTA/Yoho		10. SPONSORING/MONITORING AGENCY REPORT NUMBER DNA-TR-92-82		
11. SUPPLEMENTARY NOTES This work was sponsored by the Defense Nuclear Agency under RDT&E RMC Code B7664D SB SA LTHRR PRPD 1950A 25904D.				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) This document describes a set of artificial intelligence technologies that may help the nuclear survivability community more effectively use existing tools and data, and thus better perform survivability validation. The technologies described include expert systems, neural networks, and fuzzy logic. Potential applications to nuclear survivability problems are described for each methodology.				
14. SUBJECT TERMS Artificial Intelligence Nuclear Survivability Expert Systems Neural Networks			15. NUMBER OF PAGES 54	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT SAR	

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

CLASSIFIED BY:

N/A since Unclassified.

DECLASSIFY ON:

N/A since Unclassified.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

SUMMARY

Much attention has been recently focused on ensuring the nuclear survivability of various systems operated by the military of the United States. Traditional methods of performing nuclear survivability validation (NSV) are somewhat lacking when considering the characteristics of modern systems. This document describes a set of technologies that may help the survivability community more effectively utilize existing tools and data, and thus better perform survivability validation.

One problem with current approaches to NSV lies in the handling of uncertainty. This is evident in defining threat definitions and scenarios, predicting resultant environments, and calculating the consequent effects. Given that there are some things that simply cannot be known in advance, these uncertainties must be dealt with effectively rather than eliminated. For example, the exact geometries and weapon outputs within an engagement are unknowable. Thus, it could be argued that expending vast resources to calculate one small piece of the picture with extreme precision is probably imprudent. Other uncertainty issues exist in integrating data from various tests and/or simulations. The appropriate fusion of data from a variety of sources would render a more cohesive database of effects for NSV researchers to exercise.

Problems with uncertainty are compounded with issues regarding complexity. With complex, integrated, and redundant systems becoming the rule, traditional methods of analysis often cannot credibly predict overall system response. Many popular simulations or test methods were not designed to accurately model effects propagation from one complex subsystem to another. With new requirements such as single event upset and operate through constraints becoming important, the task of accurately performing nuclear survivability validation is complicated further.

Artificial intelligence (AI) technologies offer several methods for enhancing the ability of the NSV community to effectively deal with the problems described above. In a broad sense, AI can be thought of as using advanced computer techniques to solve problems that are generally considered to require human skill or insight. AI programs generally stress symbolic manipulation rather than numeric processing, trying to generalize rather than calculate. A number of AI paradigms exist, but the three that seem most applicable to NSV problems are expert systems, artificial neural networks, and fuzzy logic.

Expert systems are on-line consultant programs that contain the distilled knowledge of one or more experts. The expert knowledge is derived as a series of rules that the expert follows in the decision-making process. The creation of this rule base is referred to as knowledge engineering. Several potential expert systems for NSV application are described in the body of this paper, including systems for tying low-level component/material effects into analyses at the highest levels of the engagement. Other potential systems include synergistic interaction modeling, diverse data fusion/integration, man-in-the-loop modeling, and NSV protocol generation. Several issues must be addressed when building expert systems, such as expert identification and agreement, potential knowledge engineering difficulties, and updates/maintenance.

DTIC QUALITY INSPECTED 5

10m For
ORDER
AB
unread
Classification
By
Distribution/
Availability
Avail and
Dist Special
A-1

Artificial neural networks are self-training, self-organizing computer programs for modeling numeric data. As information, in the form of input-output training pairs, is presented to the neural net program, underlying trends are learned and generalities are formed. Neural nets can handle arbitrarily complex data, and thus are often able to find trends that escape human detection. Thus, handling some of the uncertainties mentioned above seems a natural application. Neural nets can also be connected to each other, via a technique called "cascading." Cascaded neural nets could be used to model complex, integrated systems in a consistent approach that would greatly benefit NSV research. Neural networks operate with great speed, but the underlying reasoning used by the nets is usually difficult to interpret. Thus familiarity with the data being modeled is mandatory.

Fuzzy logic, while not a true subfield of AI, is often applied to both expert systems and neural networks. The primary concept in fuzzy logic is partial set membership. In traditional set theory, an object is either completely a member of a set, or not at all. Fuzzy logic and fuzzy set theory allows an object to be a partial member of a set; a person could be considered seventy percent tall, for example. Fuzzy logic has been shown to be a proper superset of traditional set theory, i.e., traditional sets are a special case of fuzzy sets. This concept applies well when dealing with uncertain and/or imprecise data, such as is seen in many NSV problems.

AI research has been ongoing for about fifty years, and thus is not as recent as the popular press would imply. While the field continues to evolve rapidly, many paradigms have proven to be sound and applicable to a wide variety of problems. A rule of thumb when dealing with AI systems is that some accuracy is usually sacrificed for the sake of speed. Thus, artificial intelligence seems to be indicated where a fast, good answer is better than a slow, perfect one. Given the continual change seen in AI, care must be used to avoid problems sometimes encountered with emerging technologies. But with a consistent, coordinated approach and appropriate testing, AI techniques may present NSV researchers with a set of invaluable tools to augment their arsenal of existing models and methodologies.

TABLE OF CONTENTS

Section	Page
SUMMARY	iii
FIGURES	vi
1 PROBLEM STATEMENT	1
1.1 UNCERTAINTY AND COMPLEXITY	1
1.2 INTEGRATION	2
1.3 NEW REQUIREMENTS	3
1.4 PROBLEM SUMMARY	3
2 ARTIFICIAL INTELLIGENCE	4
2.1 AI INTRODUCTION	4
2.2 AI CONCEPTS	5
3 EXPERT SYSTEMS	7
3.1 EXPERT SYSTEMS: TECHNICAL DISCUSSION	7
3.2 POTENTIAL EXPERT SYSTEMS USES FOR NUCLEAR SURVIVABILITY VALIDATION	10
3.3 EXPERT SYSTEM ISSUES	12
4 ARTIFICIAL NEURAL NETWORKS	14
4.1 NEURAL NETWORKS: TECHNICAL DISCUSSION	14
4.2 POTENTIAL NEURAL NETWORKS FOR NUCLEAR SURVIVABILITY VALIDATION	19
4.3 NEURAL NETWORK ISSUES	21
4.4 NEURAL NETWORKS VERSUS EXPERT SYSTEMS	23
5 FUZZY SETS AND FUZZY LOGIC	24
5.1 FUZZY LOGIC: TECHNICAL DISCUSSION	24
5.2 POTENTIAL FUZZY LOGIC APPLICATIONS FOR NUCLEAR SURVIVABILITY VALIDATION	26
6 SUMMARY/CONCLUSIONS	27
7 REFERENCES	29
APPENDIX	
A AI HISTORY/BACKGROUND	A-1
B GLOSSARY	B-1

FIGURES

Figure		Page
3-1	Expert system shell components	9
4-1	Biological neurons	15
4-2	Generic neural network architecture	15
4-3	XOR function	17
5-1	Youth fuzzy set membership	25

SECTION 1

PROBLEM STATEMENT

In the event of a nuclear exchange or attack, many systems operated by the military of the United States would be required to operate in a nuclear-degraded environment. A variety of potential environments/effects, both prompt and persistent, might be encountered in a nuclear conflict. The list below enumerates some of the main nuclear effects that U.S. systems might be expected to encounter.

- Overpressure
- Ground Motion/Cratering
- Thermal
- Electromagnetic Pulse
- X-Ray Fluence
- Gamma Dose Rate
- Total Dose
- Prompt Gamma
- Neutron Fluence
- Dust
- Infrared Irradiance
- Electron Density
- Debris Gamma
- Debris Electrons
- Neutron Flux

These effects occur in a variety of altitude regimes, persist for times ranging over some twelve orders of magnitude, and have lethal ranges from a few kilometers up to global scales.

One can see from the complexity and variety of this list that the issue of validating a system to operate in such a variety of environments is a large task. When one considers the number of systems for which nuclear survivability validation must be performed, the project appears monumental. Traditionally, nuclear survivability validation (NSV) is performed in two primary ways: with hardware tests and analyses, including computer simulations. The hardware tests include underground nuclear tests (UGTs), aboveground nonnuclear tests (AGTs), and a variety of simulation devices (such as x-ray simulators). The computer analyses include first-principles physics codes, engineering/design codes, and engagement analysis codes. A third technique involves hybrid hardware/computer tests, where simulated signals that might be produced by a nuclear environment are fed into a hardware device to gauge its response.

1.1 UNCERTAINTY AND COMPLEXITY.

Using combinations of the above methods, analysts have for many years estimated the response of U.S. systems to nuclear environments. Many of these analyses have been less than robust, simply because of limited testing resources with which the studies were performed. The situation has also been subject to the effects of considerable uncertainty in terms of predicting the outputs of enemy (and friendly) weapons and their resultant environments. Details on enemy warheads are sparse, thus even if we could predict resultant environments and effects perfectly, not knowing the actual yield and output spectrum would by definition induce error. To

a lesser extent, the same is true of U.S. nuclear devices. Since full-yield testing of many systems is impossible, output characteristics must be estimated with engineering/physics models.

Similarly, with strict limits on nuclear testing, the actual responses of systems to many environments can only be estimated. While UGTs still provide the capability to perform limited nuclear environment testing, high altitude and other open-air nuclear tests are no longer available. Some data were gathered from early tests, but information on such events was rather limited by today's standards. Many large systems (aircraft, large satellites, etc.) are simply too large to be included in UGTs, so component testing of vital subsystems is the most that can be attempted. With integrated/redundant systems becoming more common, traditional methods of analysis are often not able to believably predict overall system response. Most first-principles codes were simply not designed to model effects propagation from one complex subsystem to another.

A solid understanding of subsystem response is important when considering the widespread use of embedded systems that are common to more than one weapon system. For example, if a common fuzing system is used by several missiles/warheads, an error in assessing its nuclear survivability might affect the viability of several vital systems. This problem is especially important for "smart" weapons that the U.S. relied on so heavily in Desert Storm. If a "smart" bomb that relies on accuracy to allow low warhead yield loses its ability to home on a target, it is a much less valuable weapon. We have not yet encountered a conflict in which these weapons were forced to perform in a nuclear-degraded environment.

1.2 INTEGRATION.

Given the wealth of data that would need to be produced to accurately perform NSV, data integration also becomes an issue. Similar data have been acquired in a variety of tests. Redundant information from two or more UGTs requires correlation. Given inexact test procedures and varying test conditions, conflicting data may also emerge, requiring resolution. Similar problems can arise with AGTs and simulators. Uncorrelated or conflicting multi-test data can (and have) lead to disagreement in the NSV community regarding which results should serve as the metric. A further complication of this problem is cross-correlating data from distinct test types. For example, data from AGTs, UGTs, and simulators may exist for certain effects. Given the distinct nature of these test methods, it would not be surprising to see differences in the results. However, these diverse data must somehow be reconciled.

Test data is not the only area in which conflicts arise. In many cases, NSV is performed largely with computer code simulations. As with various tests, various codes often produce dissimilar results, sometimes differing by orders of magnitude. Sometimes this arises from using different engineering approaches, assumptions, boundary conditions, or from a different method of applying first-principles physics. Often, the codes have been adjusted to corroborate the results of a test, and codes calibrated to different tests cannot be expected to return the same answer.

Finally, attempting to compare the results of predictive computer codes with the results of actual tests is often an inexact science. Poorly understood physics and computational simplifications are often to blame, as is unexpected weapon output variation. In order to have a believable, cohesive set of NSV protocols and regimens, test data and code results must be reconciled, and proper predictive tools identified.

Another factor in NSV data integration lies in integrating information for multi-level system interactions. Data for large or complex systems would have likely been produced at the lower levels of integration, such as the piece-part or materials level. These results must be formed into a cohesive model for major subsystems, and then further into platform structures. Often, especially in global systems such as satellite networks or ABM systems, these individual platform models must be incorporated into a "system of systems" level analysis. Care must be used when performing this multi-level integration since errors at the low levels propagate through the analysis to the highest levels. Error budgets must be assigned and error analysis must be performed to ensure models that reflect proper use of the available test and simulation data.

1.3 NEW REQUIREMENTS.

Recent analyses have shown a need for U.S. systems to survive in certain circumstances that were not considered a decade ago. Single event upset (SEU) is one example of an effect that has gained recent attention. SEU occurs when sub-atomic particles pass through computer memory circuits, causing "bit flips" that can degrade or stop on-board processing. Electrons generally do not penetrate deeply enough to cause SEU. Additional considerations when assessing for NSV in terms of SEU include circumvention algorithms in addition to the usual materials responses.

Closely related to the issue of SEU is that of "operate through" requirements. Applicable systems are required to operate through severely nuclear-degraded environments without being forced into circumvention. In these cases, new electronics hardening/shielding techniques must be considered, the understanding of which may not be as mature as for less demanding systems. Other new requirements are certain to emerge as systems are proposed and designed to be survivable in nuclear environments.

1.4 PROBLEM SUMMARY.

It has been shown that, with the current emphasis on nuclear survivability and its validation, traditional methods of assessing nuclear effects have some shortcomings. Deficiencies exist in terms of dealing with uncertainty and complexity. Unresolved issues are also evident for integrating test data and computer simulation results. New requirements demand the acquisition of new methods of analysis. It is a credit to the NSV community that traditional methods have been used as successfully as they have over the past decades, and discarding current tools and methodologies would be folly. However, new technologies are emerging that may well hold the key to using these tools more effectively, efficiently, and accurately. The remainder of this paper will deal with one such set of technologies.

SECTION 2

ARTIFICIAL INTELLIGENCE (AI)

Artificial Intelligence (AI) is the collective term for a set of technologies computer scientists have developed to solve problems using non-traditional programming practices. Whether AI programs actually exhibit or possess intelligence or not is a debate best left to the combatants, and the issue is unimportant for the purposes of this paper. For semantic purposes and convenience, the phrase "AI" will be used here assuming none of the philosophic trappings. There is no single widely-accepted definition of AI, but in this paper, we will assume that AI involves using non-traditional programming practices and languages to solve problems that are generally considered to require human expertise or insight to solve, or to solve them in a way similar to the human brain. These problems and associated techniques are widely varied, and no single paradigm defines or is common to all subfields of AI.

2.1 AI INTRODUCTION.

While AI has been common in the popular press only for about a decade, AI's roots reach into the 1940s (McCullogh, 1943). Early research was conducted by physiologists interested in the inner workings of vision and learning. Work continued and migrated towards computers in the 1950s, and the field blossomed in the 1960s. During the 1960s, early AI successes with artificial neural networks (described in Section 4 below) in very limited problem domains prompted great speculation and, in turn, investment. When neural nets failed to provide a panacea for most computing problems, AI researchers with competing technologies quickly criticized them, and promoted their own projects (Minsky, 1969). Disillusioned investors turned away from AI until the late 1970s, when the expert systems (described in Section 3 below) boom began.

The current AI push began with the demonstration of some powerful, laboratory-built expert systems that solved some narrow problems that had previously been considered solely human expert territory. In their narrow domains, these systems often performed their tasks as well as the best humans. The Japanese government initiated their "Fifth Generation Computer Project" in the early 1980s, planning to create a general purpose, natural language computer. Their plan was to use rule-based programming in the ProLog language to achieve their goal. The U.S. response to the Japanese project was dual. The Defense Advanced Research Projects Agency (DARPA) started the Strategic Computing Initiative for developing several advanced computer technologies for (mainly) military application. U.S. industry responded by creating the Microelectronics and Computer Technology Corporation (MCC), a consortium of high-tech American firms to pursue similarly lofty goals. The Japanese project proved unachievable, and has redefined its scope significantly to match the project's limited achievements. Several considerable technology advancements have emerged from the MCC, but it too has undergone significant retrenchment. The Strategic Computing Initiative produced several specialized applications, some of which have found their way into deployed U.S. systems. However, there has been little transfer of the DARPA-funded technology to non-military applications.

The 1980s saw a huge commercial investment in AI, particularly in the area of expert systems. For reasons discussed below, expert systems are no longer held in the favor they were five years ago. Of the hundreds of start-up expert systems software and consulting firms that appeared in the early 1980s, only a handful

remain. While there is still quite a bit of activity in the expert systems arena, it has become a "niche" market compared to what was expected a few years ago. The AI boom prompted by expert systems would have ended during the last few years were it not for a 1986 breakthrough in neural network technology that has overcome many of their shortcomings of the 1960s. Current commercial AI research now focuses largely on neural networks.

2.2 AI CONCEPTS.

One of the main ways in which AI differs from traditional computer programming is that while mainstream techniques focus on numeric processing, AI primarily uses symbolic processing. The most-used programming language for AI is called LISP (LISt Processor). LISP programs consist completely of lists of objects and operators to manipulate them. These objects consist of names, facts, and other objects, and the operators perform such activities as list maintenance and object comparison and evaluation. These objects do include numbers, so primitive arithmetic functions may be performed, but the language is so tailored to non-numeric computing that an equal sign is not a part of LISP's syntax. An exception to this AI trend is that neural networks require a module to perform quick numeric matrix manipulation.

Given this symbolic emphasis, it should come as no surprise that tasks requiring numeric speed and accuracy are poorly suited for LISP application. If, however, the task can be modeled symbolically rather than numerically, AI programs often provide impressive decreases in computer run-time. It is not uncommon, for example, for a neural network to operate many orders of magnitude quicker than corresponding numeric programs. How this is achieved with expert systems and neural networks will be discussed in detail later. Another general rule with AI programs is that when these speed gains are achieved, the price is usually paid in accuracy. Since most AI programs attempt to generalize rather than calculate, most results are estimates of reality. In short, AI applications usually have the most to offer when a good, quick answer is more desirable than a slow, perfect one.

There are many subfields of AI currently being researched. Most of these are known and studied only in the academic community, but several have emerged as commercially viable in industry. The three main topics of interest in terms of NSV are expert systems, neural networks, and fuzzy logic. These will be introduced here and detailed in subsequent Sections. Some other AI subfields will be listed briefly.

Expert systems are programs written to distill and apply the knowledge of human experts in narrow expertise domains. The abilities of the expert are formed into a series of IF-THEN rules by an AI programmer called a knowledge engineer. These rules are then incorporated into an expert system "shell" program which is used to manage and apply the rules as needed. The user of the system inputs information on the problem at hand, and the expert system asks for more data as is necessary and outputs its diagnosis of the situation. The user can then ask the system to describe how the conclusion was derived, and the expert system will list the rule chain and fact set that led it to the response. Expert systems have been created to solve such problems as disease diagnosis, optimal computer configuration, and battlefield assessment.

Artificial neural networks are programs that take a set of data in the form of corresponding input-output pairs, and analyze the data for underlying trends and patterns. After learning these patterns, input data which the neural net has not seen can be presented, and the network will give its estimate of the appropriate

output. Unlike the expert system, very limited information on how the neural net arrived at its conclusion is available. Neural networks have been created to solve problems such as financial prediction, weather forecasting, and hypervelocity projectile lethality.

Fuzzy logic is a technology that allow objects to be classified and acted upon without precise knowledge about the objects. In particular, fuzzy set theory defines the concept of partial set membership (fuzzy sets) and fuzzy set operators. For example, traditional ("crisp") set theory would allow a set to encompass all tall people. An arbitrary discriminant would be required to determine set membership requirements; six feet, two inches or taller, for example. The set of short people might include those five feet or shorter. Medium height would include all others. The implicit problem is that sharp boundaries must be imposed for imprecise concepts. Fuzzy logic would allow the construction of two sets where, for example, a person five feet, ten inches would be considered 70 percent tall and 30 percent short. Fuzzy set operators (union, intersection, etc.) have been defined to allow familiar set manipulations of fuzzy sets. Fuzzy logic has been applied to problems such as automatic camera focusing, financial/insurance risk assessment, and database information retrieval.

Other subfields of AI that have generated considerable interest include intelligent database search methods, computer vision and other robotics fields, natural (human) language understanding, computer speech, intelligent tutoring and training systems, rapid prototyping, and computer gaming (chess programs, etc.).

SECTION 3

EXPERT SYSTEMS

ELIZA is a simple parsing computer program that was popular in the 1970s (Weizenbaum, 1965). Although it is little more than a series of IF-THEN and text string manipulation statements written in an obscure computer language called MAD, it acts like an interactive Freudian psychotherapist. A session with ELIZA usually leads to a breakdown of its parsing and sentence reconstruction algorithms, producing nonsensical and often humorous responses to the user. ELIZA will carry on conversations with the user indefinitely, prompting for more information on certain keywords that the user had mentioned in the course of the dialogue. However, never reaching a solution to the user's problem does present some element of realism. While it was created as little more than a toy, it set the stage for the legitimate computerized experts that AI was about to produce.

3.1 EXPERT SYSTEMS: TECHNICAL DISCUSSION.

Expert systems are also often referred to as knowledge-based or rule-based systems, as the primary data structure is generally the IF-THEN rule. Rather than writing and debugging a program, expert system creation consists mainly of deriving and elucidating rules and knowledge. With expert systems, the emphasis is on what to know rather than what to do. The IF side of the rule is called the antecedent, and the THEN part is the consequent. A rule from MYCIN (Shortliffe, 1976), an expert system for medical diagnosis, is shown below:

```
IF  the infection is primary-bacteria
    AND  the site of the culture is one of the sterile sites
    AND  the suspected portal of entry of the organism is the
        gastrointestinal tract,
THEN there is suggestive evidence (0.7) that the identity of the
    organism is bacteroids.
```

In the above rule, there are multiple antecedents. It is also common for a rule to have multiple consequents. The antecedents here are connected with logical AND operators. Other logical operators used in the antecedent include NOT (negation), OR, and XOR (exclusive OR - this OR that, but not both). The only legal logical operators for use in the consequent are AND and NOT. In the consequent shown above, there is a confidence factor of 0.7 shown, indicating that if the conditions in the antecedent are met, there is a 70 percent probability of the consequent being true. There are special operators internal to expert systems that propagate the probabilities through multiple layers of rules to determine a final diagnosis confidence factor. These are important, as many expert systems will output a list of possible conclusions, each with a probability attached, rather than a single diagnosis.

The process of deriving the rule base for the expert system is called knowledge engineering. This process includes interviewing the expert (or analyzing some database) to determine the appropriate issues or parameters. The relationships between these issues is investigated next, and the prototype rules are formed. This rule base is then checked for completeness and consistency, and as necessary, the knowledge engineering process iterates. The knowledge engineer must also determine the appropriate structure for the rule base, and the best inferencing technique (described below). While this method sounds straightforward, unsuccessful

knowledge engineering was a large factor in the recent expert systems decline (see Appendix A).

The rules are manipulated within an expert system with a part of the program called the inference engine. This module applies the appropriate rules when needed, and searches for facts concerning the current situation. These rules and facts are combined by the inference engine to determine the diagnosis or solution. The expert system can also explain to the user why it wants to know a certain piece of information or list the rule path that led to the answer given. There are two main inferencing techniques used by expert systems: forward and backward chaining. Forward chaining takes a (hopefully) complete set of facts and works toward possible conclusions, while backward chaining picks a possible solution and attempts to prove or disprove it based on the facts available. In theory, both should give the same answer, but getting to the solution may be easier for one mode given a certain class of problems. To show how forward and backward chaining work, consider the following "expert system" and set of facts.

Rules:

- (1) IF a THEN b.
- (2) IF b THEN c.
- (3) IF c AND d THEN e.
- (4) IF c AND NOT d THEN f.

Facts:

a = TRUE.
d = FALSE.

The system would have the additional information that the desired answer is whether e is TRUE or if f is TRUE. In forward chaining, the sequence of events within the inference engine would go something like as follows. Given that a is TRUE, b is TRUE from (1). The process of proving the antecedents of a rule to "fire" the consequent is called instantiation. Since b is TRUE, c is TRUE from (2). Since d is FALSE, (3) does not apply. Since c is TRUE and d is FALSE, f is TRUE from (4). The system would then determine that there are no further applicable rules and report that f is TRUE (if asked about the status of e, the response would be UNKNOWN).

With backward chaining, the system would first assume e is TRUE. It would then scan the rule base for rules with e set to TRUE in the consequent, and consider the premises of the antecedent. In this case, it would find that for e to be true, c and d would need to be true from (3). Checking the facts, it would find d to be false, and thus (3) is not applicable. It would look for other rules with e in the consequent, but finding none it would abandon e and repeat the above algorithm for f. In (4), it would find that c needs to be TRUE and d needs to be FALSE. The fact base would show d to be FALSE, and no information about c exists. Thus the rule base is searched for rules with c in the consequent, finding (2). Since the status of b in (2) is unknown, the inference engine would continue to chain backwards, finding (1) with b in the consequent. Here, though, it finds resolution, with a = TRUE being in the fact base. This result is then propagated back up the chain, determining f to be TRUE. Again, the status of e remains UNKNOWN.

For classes of problems where there is a relatively small number of conclusions, backward chaining works quite well, and quickly. Conversely, for problems with a large (or infinite) number of answers, forward chaining is indicated. Most commercial packages automatically use whichever inferencing technique is indicated, often using both within a single consultation. In an actual expert system, there would be many more rules (tens to thousands) than were shown in the example above, and the fact base would be created dynamically during the run, with the

expert system prompting the user for information. It should now be more clear how important a correct and complete rule base is to expert system development.

There are several parts of an expert system besides the rule base and inference engine, but they are generally support modules. Figure 3-1 below shows the parts of a generic expert systems shell (Lenat, 1984). In this figure, the rules are the only part which are necessarily created by the expert system designer/knowledge engineer. The remaining parts of the system may be created for an individual expert system, but this is usually not the case. The various support modules are available as part of several commercial expert system packages. Most packages include considerably more capabilities than the shell outlined above, including such modules as industrial process control tools, graphical shells, and multi computer language interfaces. Figure 3-1 shows the parts of a rather Spartan, traditional expert system shell.

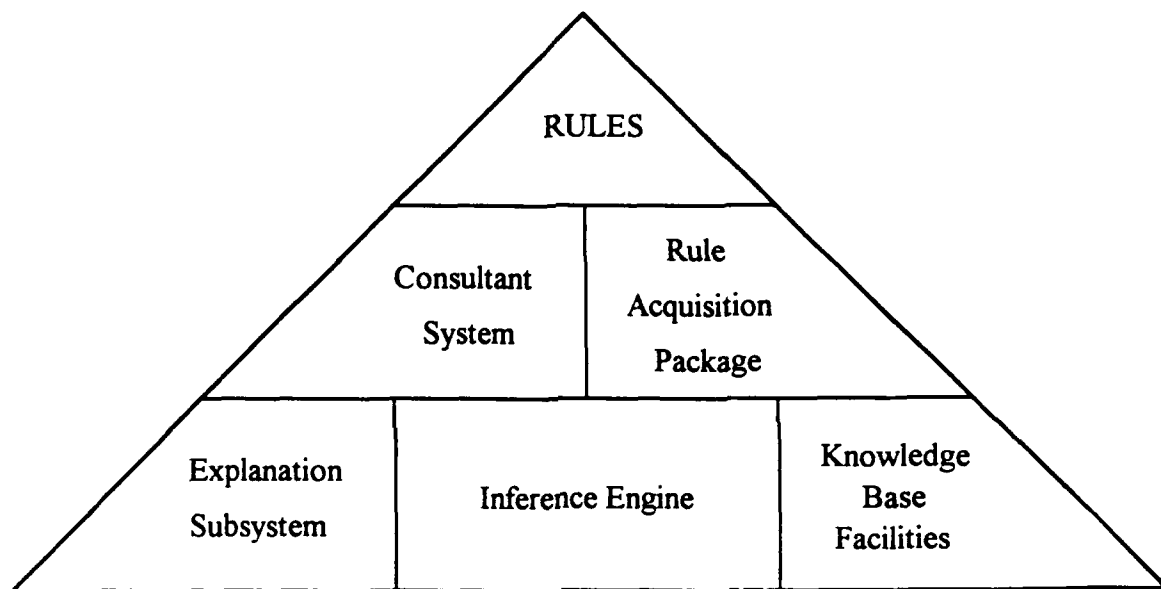


Figure 3-1. Expert system shell components.

In Figure 3-1, the consultant system asks the user for the status of various parameters needed by the inference engine. The rule acquisition package is the part of the program wherein the user inputs and tests the rules. It also prompts the user for rules on certain parameters for which it judges insufficient rules exist. The explanation subsystem gives the user information regarding why it wants to know a certain fact, the line of reasoning it is currently pursuing, and the rules that were used to reach the final conclusion. The knowledge base facilities are routines that check the rule base for consistency. For example, two sets of rules that are seemingly consistent may actually lead to a logical contradiction (i.e., implying something and its opposite). In large rule bases, it is common for contradictory rule combinations to appear, and they usually lead to severe problems if not detected.

3.2 POTENTIAL EXPERT SYSTEMS USES FOR NUCLEAR SURVIVABILITY VALIDATION.

Given the NSV problems outlined in Section 1, there are several ways in which expert systems might be used. Consider the problems described with integrated system component interactions. An expert system might be constructed to help propagate low-level effects to determine system impact. For instance, a rule from an expert system for nuclear warhead modeling might be something like:

IF circumvention has occurred since reentry
AND altitude sensor is inoperational,
THEN warhead fuzing mode is impact
AND height of burst is 0
AND warhead lethality radius is diminished by 40%.

Rules such as this would serve to tie low-level component/materials effects into analyses at the highest levels of the engagement, without extensive numeric and engineering calculations. Rules would be derived from experts in the lethality and engagement fields, as well as from effects and materials databases in existence. Various rules would be designed to consolidate all levels of interaction: materials, piece-part, component, subsystem, system, and system of systems. Similarly, complex systems of systems might have several different systems incorporated (sensors, weapons, communications, etc.). An expert system could be designed to model the synergistic interactions between vastly different systems. A possible rule for an ABM effectiveness expert system might read:

IF nudet has occurred uprange less than 50 kilometers
AND sensor type is ground-based phased array radar,
THEN target warhead detection range is degraded by 20%
AND maximum interceptor range is degraded by 30%
AND maximum target commit altitude is 600 kilometers
AND minimum intercept altitude is 75 kilometers.

Please note that the rules and values contained therein are entirely fictitious, but representative of what could be expected in actual expert systems.

Another problem area described in Section 1 was that of integrating data from various sources and resolving conflicts between them. Recall that these conflicts occur among simulation codes, among various tests, and between codes and tests. An expert system could be constructed, based on NSV community consensus, that would define what results are to be considered the baseline in different regimes. For example, a rule from such an expert system might read:

IF burst altitude is exoatmospheric
AND affected system is GPS satellite,
THEN use results of HOLY SMOKES event
AND use results of ULTRANUKE code.

Again (obviously), this rule is a fictitious example of what might actually be used.

A third area in which expert systems could potentially aid in nuclear survivability validation is effects/response characterization. An expert system might be constructed to distill the results of effects and response codes to give estimates of their outputs. For example:

IF burst altitude is exoatmospheric
AND range is less than 30 kilometers
AND x-ray fluence is greater than 0.75 cal/cm²,
THEN category 3 electronics exhibit type 3, 14, 22 effects
AND category 6 electronics exhibit type 3, 14, 22, 28 effects.

Extensive work in categorization would be required to construct such an expert system, but another expert system could possibly help out there, too. Other rules could include scaling equations to determine specific failure probabilities or individual system effects.

In addition to the problems outlined in Section 1, there are several other areas in which expert systems could be applied to nuclear survivability validation. For instance, consider the task of creating nuclear survivability validation protocols and regimens. This task entails determining the efficacy of various tools to assess effects, and defining the ways in which they should be used. These protocols would be created for a very wide range of systems. However, many of the systems would have either common components or very similar ones. There are probably very few U.S. systems that are completely dissimilar to all others. Thus the expertise used for NSV for one system could, in most cases, be applied to another. In fact, if an inconsistent approach (such as might emerge from creating NSV protocols without common tools/guidelines), would likely lead to an inconsistent set of protocols that would not be considered credible in the nuclear survivability community. An expert system, or more likely a set of expert systems, could be constructed to be used by protocol writers to help maintain consistency. These expert systems would require information on components, configurations, materials, and other pertinent information, and would counsel the protocol analyst on a recommended set of tools and NSV methodologies. A sample rule might be something like:

IF system is ground-based interceptor
AND guidance mode is inertial
AND subsystems of interest is ACME RLG,
THEN recommend code group 4 for neutron flux analysis
AND recommend code group 7b for x-ray fluence analysis
AND recommend NOT use of event 23 thermal results
AND recommend database 16 for EMP analysis.

Again, this is a contrived example, but would likely be representative of the type of rule that would be needed to construct an expert system to aid in NSV protocol development. Distinct expert systems would likely be needed for various types of systems, and perhaps even for various effects, in order to keep the expert system sizes manageable. In such an event, it would be important to maintain consistent rule bases among the expert systems.

Expert systems would also have a place in engagement modeling used for NSV, especially at the systems or system of systems level. Many engagement codes currently ignore the effect of "men-in-the-loop." Others simplify the effect by modeling a time delay and a fixed probability of making the correct decision. A higher fidelity model could be created using an expert system to model the human effect. An expert system could be constructed to make the human decisions in much the same way (and with roughly the same accuracy) as a man-in-the-loop would. The expert knowledge imparted to the expert system would allow the program to not only make mistakes with more accurate frequency, but would also tend to make

the correct mistakes (i.e., the same mistakes the human counterpart would likely make). Thus the effect of the man-in-the-loop and potentially incomplete information could be better gauged. An estimate could also be made of the time required to make the decision based on the number of rules that fired (or perhaps the number of antecedents and consequents to account for rule complexity). Alternatively, the run-time of the expert system could be scaled to account for the time delay. This methodology might be one of the only accurate ways to assess the effect of humans on nuclear survivability.

This section has presented several potential applications for expert systems in the nuclear survivability validation arena. This list is by no means complete or exclusive. As technologies such as this are demonstrated, the need for new applications usually emerges. With these examples, the concept of how and where expert systems might be used was demonstrated. The next section attempts to convey where they may be inappropriate.

3.3 EXPERT SYSTEM ISSUES.

While expert systems technology offers a powerful, non-traditional way to approach many difficult problems, it is not a panacea. In fact, the over-hyping of expert systems in no small part led to a recent shakeout in the expert systems industry. Having a good grasp on the issues that must be addressed before entering in to the construction of an expert system is vital to success of the project.

The first obvious issue in attempting an expert system is identifying the expert that is to be modeled. Traditionally, this expert is a person (or persons), but expert-less expert systems have also been constructed from databases or other existing sources of information. The point is, the expertise must exist somewhere before it can be modeled. Expert systems are not the correct tool for attempting to invent expertise where humans have failed, although this may produce new insights. The expertise must be available, and if human, agreeable. In numerous cases in the past, individuals have been less than willing to participate in the automation of their knowledge, fearing it would diminish their status or make them superfluous. From the viewpoint of the AI scientist, it is an honor to be considered expert enough to warrant the effort of computer emulation. To those unfamiliar with AI and expert systems, it may seem to be a trivialization of their abilities.

Another issue regarding the choice of experts regards potential disagreement among experts. Most experts have very strong ideas that they will not allow to be challenged. Human nature, however, teaches that rarely would two experts in the same field agree on all issues. If multiple experts are being used, the knowledge engineer will likely have to arbitrate disagreements between experts, or simply choose an opinion to use. There is also the possibility that an outside expert will challenge the system with issues either legitimate or frivolous. Even if a computer code or database is used as the source of the expertise, proponents of competing or opposing products may cause similar arguments. Given that experts in the same field often disagree, experts must sometimes simply be wrong. Not surprisingly, this effect can propagate into the expert system.

As was mentioned above, the most difficult task in creating an expert system is the knowledge engineering. While uncooperative experts occasionally present difficulties, more often the most serious problems arise when the knowledge engineer and the expert are unable to derive the knowledge that underlies the expert's abilities. The problem seems to arise almost from the very definition of an expert.

It has been said that "an expert is really no smarter than you or I; he is just better organized and has viewgraphs." Were this the case, expert system generation would be a much more straightforward task. In reality, a person achieves expert status upon development of the ability to solve problems or answer questions without having to go through extensive mental machinations.

An analog for this exists in the computer science world. Consider a computer program written in FORTRAN, Pascal, or any other compiled language. The source code is readable by any programmer fluent in the appropriate language. Once the program is compiled into an executable file, it is impossible to get any useful information about how the problem is solved by looking at the executable file. Tools such as disassemblers are available for performing a limited translation back to understandable code. However, once compiled, the only good way for the executable code to yield knowledge is by actually running the program. It appears that most true experts have, over a number of years, "compiled" their knowledge much like in the above description. The ability to solve the problem exists, but often the expert does not know exactly why the answer is. It is the task of the knowledge engineer to "disassemble" the expert's "program" back into "source code" that can be used in an expert system.

Another knowledge engineering difficulty lies in the fact that in order for the knowledge engineer to successfully derive the rule base, he must have a reasonable understanding of the expert's field. It would be very difficult for an accurate and complete rule base to emerge if the knowledge engineer doesn't know the right probing questions to ask. It is not as simple as asking the expert, "So just how do you do this, anyway?" Knowledge engineering is a highly iterative process wherein as the knowledge engineer's understanding of the expert's field sharpens, the appropriate issues become clearer. A good example of this occurred in the construction of the MYCIN (Lenat, 1984) expert system. As a byproduct of creating MYCIN (an expert system for diagnosing and treating infectious blood diseases), knowledge engineer and system developer Edward Shortliffe became Dr. Edward Shortliffe, M.D. As long as the knowledge engineer has a reasonable proficiency in the problem domain, the level of understanding necessary to construct the expert system will usually evolve.

Another issue that must be addressed before starting to build an expert system is that of updates and maintenance. When the expert system is first fielded, especially if it is widely distributed, certain shortcomings will likely emerge that were not found during development and testing. Also, as the problem domain changes, or as new tools or data become available, provisions must be made to allow the expert system to grow or change to reflect this. The point here is that the utility of a static system is usually marginal. It should be expected that to maximize the usefulness of an expert system, it should be viewed as an ongoing effort. For example, the R1 (McDermott, 1980) expert system for configuring DEC VAX computers has grown by about an order of magnitude since its first application.

SECTION 4

ARTIFICIAL NEURAL NETWORKS

Science fiction has long regaled us with tales of intelligent computers and robots with human-like abilities and personalities. To computer scientists, such machines have been little more than dreams, for it has long been clear that there are strict limits on the abilities of programs that consist of "In situation X, do Y." Databases of instructions could never be complete, and would likely fail to identify correct contexts quite often anyway. It is clear that if a machine is to emulate man, it must learn and process information like man. In this light, recent developments in the field of neural networks have presented the most promising existing technology for creating computers that at least seem to use the same type of internal data structures as humans.

4.1 NEURAL NETWORKS: TECHNICAL DISCUSSION.

Neural networks are the sole successfully deployed AI paradigm that attempts to mimic the activities of the human brain and how it physically operates. Expert systems, while they mimic the abilities of the brain, use a data structure and algorithm that was contrived largely for the sake of convenience. The primary primitive data structure in the human brain is the neuron. There are approximately 10^{11} neurons in the human brain. Extending from the neurons are tendril-like axons, which carry electrochemical signals from the body of the neuron. Thinner structures called dendrites protrude from the axons, and continue to propagate the signals from the neural cell bodies. Where the dendrites from two neurons meet, interneural signals are passed. These intersection points are called synapses. Figure 4-1 shows an extremely simplified representation of two connected biological neurons.

Connecting the 10^{11} neurons are about 10^{15} synapses. According to neuro-anatomical theory, the human brain "works" through the continual electrochemical stimulation of groups of neurons. When a neuron receives a threshold amount of electrochemical stimulus, it "fires" a signal of its own, the magnitude of which seems to be a function of the sum of all incoming signals, and a cascading effect occurs. Learning theory has determined that when two connected neurons fire simultaneously, the connection between them is strengthened. Since each neuron is connected to many others, it is clear from this architecture and operation how a single neuron may be involved in several functions/memories. Depending on which adjacent neurons are firing, a neuron will send out various signal levels. Human memories and mental abilities appear to lie largely in the connectivity and synaptic organization of the brain. Exactly how the human brain "trains" and organizes itself is still somewhat of a mystery to biological researchers.

Artificial neural networks are computer programs that emulate some of the higher-level functions of the architecture described above. As in the human brain, there are neurons and synapses modeled, with various synaptic connection strengths (referred to as weights) for each connected pair of neurons. However, similar to many computer programs (and unlike the brain) there is a specific set of input and output neurons for each problem and each net. These input and output neurons correspond to the input and output parameters of a traditional computer program, and the other neurons, along with the synapses and their weights, correspond to the instructions in a standard program. Figure 4-2 shows a representation of a generic

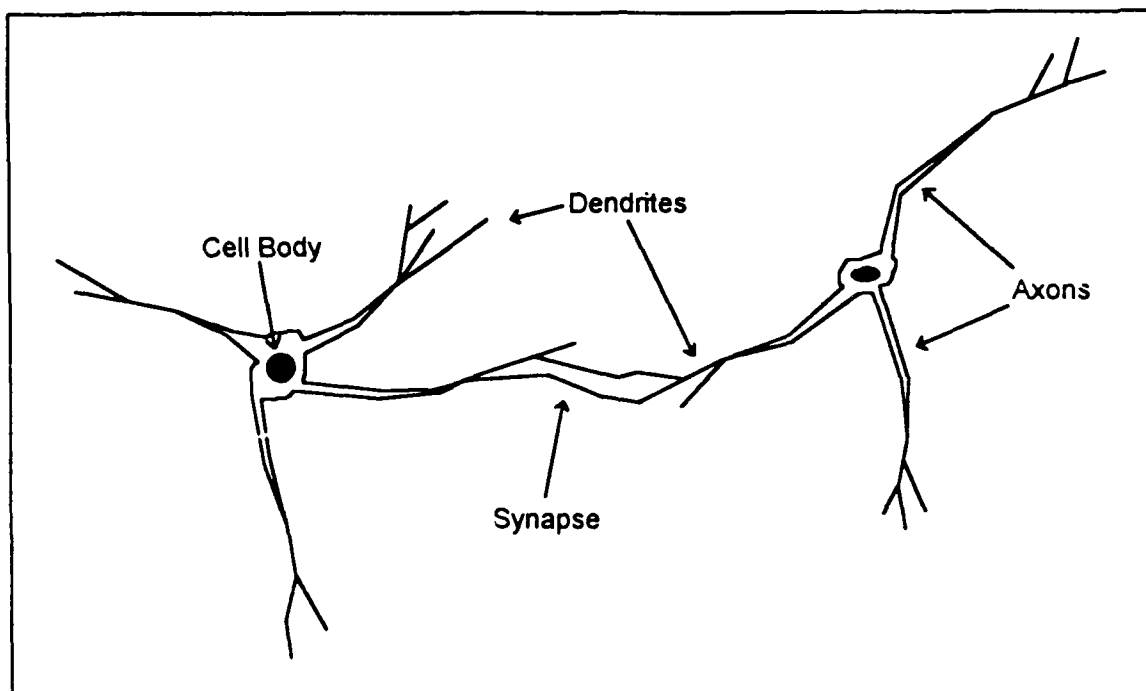


Figure 4-1. Biological neurons.

artificial neural network. The network shown has seven input neurons, two output neurons, two hidden layers, nine hidden neurons, and 63 synapses (and weights).

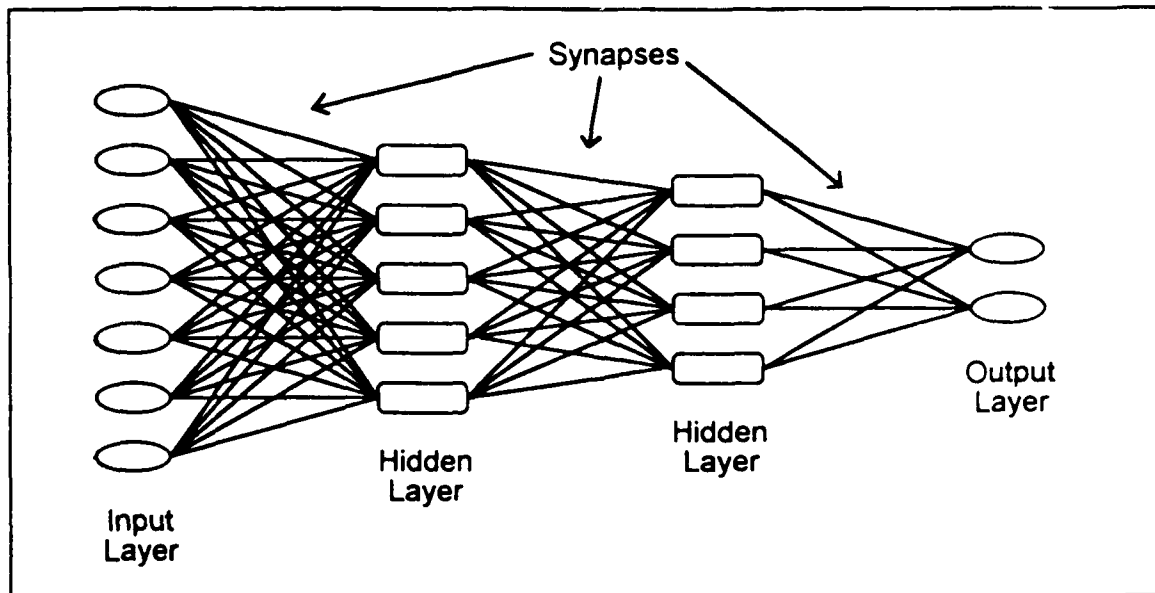


Figure 4-2. Generic neural network architecture.

Notice that the network is organized into layers. There are four layers of neurons and three layers of synapses in the network diagrammed in Figure 4-2. Some researchers would refer to this as a four-layer network, while others would call it a

three-layer network. Since the synaptic weights contain the main information in the system, in this paper the latter nomenclature will be used. Thus the network in Figure 4-2 will be considered a three-layer network. While the purposes of the input and output neurons were described above, the hidden layers may seem confusing. They are called hidden since users do not explicitly define the values contained in them, rather they are filled as a byproduct of the calculations involved in running the network. Also, the hidden neuron contents are generally intermediate calculation results that have little or no apparent meaning in most cases. Thus since they are generally not examined by users, they remain hidden from view.

Synapses exist between all neurons of adjacent layers. Weights are defined for each synapse, to promote or inhibit the simultaneous firing of connected neurons. These weights are collected into matrices, one for each layer of synapses. Thus the matrix containing the weights of the synapses between the input neuron layer and the leftmost hidden neuron layer would be a seven by five matrix. To run the network, a given set of seven input values (the input vector) would be multiplied by this matrix, yielding a vector for the first hidden neuron layer. This vector would have five elements, each one representing the sum of the seven signals incoming from the input layer. Each of the seven signals incoming to each of the five neurons would simply be the appropriate input neuron value multiplied by the corresponding synaptic strength.

At the hidden layer, the signal sums are modified before being passed on. Since neurons with large incoming signal sums would tend to overwhelm those with small incoming signal sums, the sums are generally passed through an activation function that compresses the sums into smaller numeric range, often between zero and one. This function also makes all large sums look about the same (function output of about one) and makes all small sums look about the same (function output of about zero). There are several functions that are used to do this. One of the earliest-used was a linear function, where sums below or above some thresholds are mapped to zero or one, respectively, and between the thresholds the function assumes a straight line. A more powerful function is the sigmoid function, which makes very small or large sums look similar (near zero or one), but in the one-sigma part of the distribution, an S-shaped curve completes the map. Activation functions that compress incoming signal sums into a set range like this often referred to as squashing functions, and neurons that incorporate them are called squash neurons. Other squashing functions include step functions, hyperbolic tangents, and gaussian functions.

Continuing the example neural net in Figure 4-2, the squashed values from the leftmost hidden neuron layer would then form the input vector for multiplication by the five by four weight matrix between the two hidden neuron layers. The resulting four-element vector at the rightmost hidden neuron layer would then be squashed and multiplied by the four by two weight matrix between the rightmost hidden layer and the output neuron layer. The final resulting two element vector would then be the output of the neural network for the given set of input neurons.

Early neural networks were single layer systems called perceptrons (Rosenblatt, 1958). Training perceptrons (that is, adjusting the synaptic weights so that they solve a problem) was a relatively simple task. A training algorithm was derived that was proven to be able to train a network to solve any perceptron-solvable problem (Rosenblatt, 1962). The problem was only a few interesting and useful problems could be addressed with perceptrons.

For instance, one famous example of a function that perceptrons could not model was the logical exclusive OR. Recall that the XOR function returns the value of TRUE if one of its two operands is TRUE, and FALSE if none or both are TRUE. Consider Figure 4-3 that shows the outputs of $f(x,y) = x \text{ XOR } y$. In this graph, 0 and 1 represent FALSE and TRUE, respectively.

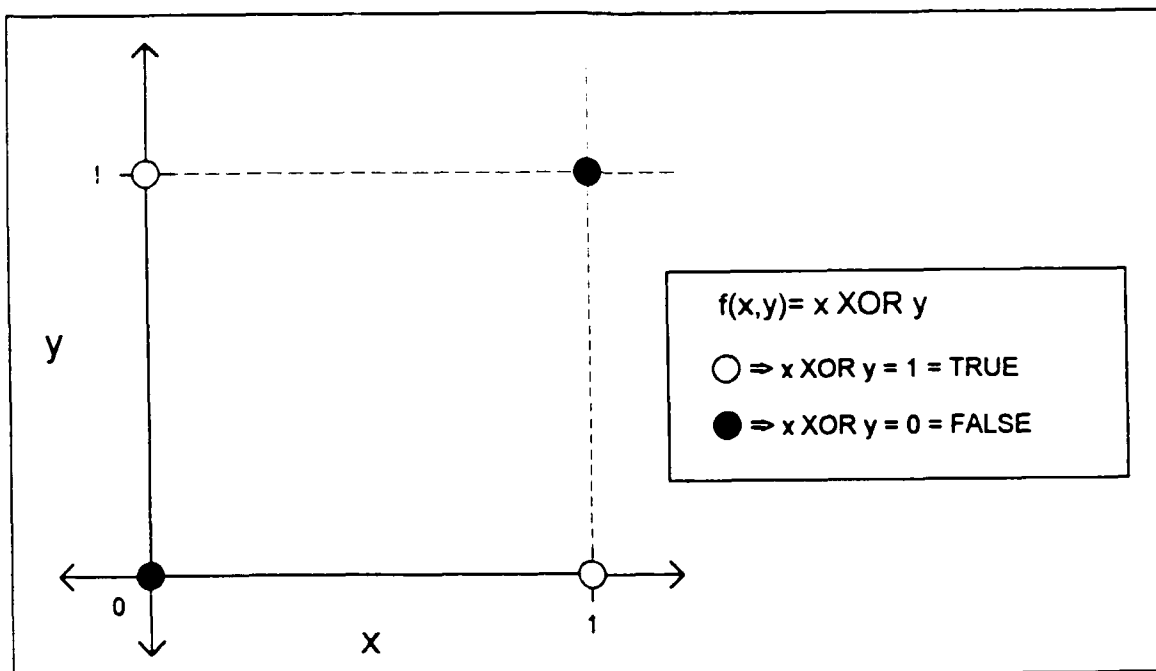


Figure 4-3. XOR function.

It was proved that perceptrons can only solve problems that are linearly separable. For the XOR case shown in Figure 4-3, that means in order to be modeled by a perceptron, one must be able to draw a straight line on the graph that divides the plane such that all like values lie on their own side of the line. A cursory inspection of this figure shows that no straight line can be drawn to separate the hollow circles from the filled circles. For problems with more dimensions than this simple example, a three-dimensional problem space would need to be separable by planes, a four-dimensional space by hypercubes, and so on.

The development of the backpropagation training algorithm (Rumelhart, 1986) finally presented an effective and reasonably efficient method for training multi-layer neural networks. This algorithm removed the largest roadblocks to neural net utility that had appeared in the 1960s. Neural network training, by backpropagation or otherwise, is similar (on the surface, at least) to human learning. Artificial neural network training is done basically by presenting the network with sets of inputs along with their resultant outputs. These data sets are called training pairs, and they consist of the known corresponding input and output neuron vectors as described above.

The initial network configuration (number of hidden neuron layers, number of hidden neurons in each layer, activation function, training rate, error tolerance) is chosen by the system designer. There are no set rules to determine these network

parameters, and trial and error based on experience seems to be the best way to do this currently. The synaptic weights are randomized, so that the system initially consists of "white noise."

The training pairs are then run through the network as described above, to see how many cases it gets correct. A correct case is one where the input vector's network result is sufficiently close to the established output vector from the training pair. Initially the number of correct cases will be very small. This is referred to as a forward pass through the network. The network training module then examines the errors and adjusts the synaptic weights in an attempt to increase the number of correctly assessed training pairs. This adjustment is referred to as a reverse pass through the network. Once the adjustments have been made, the training pairs are again presented to the network, and the entire process iterates. Eventually, the number of correct cases will reach a maximum, and the iteration can end.

Once the training is complete, testing is performed. When creating the database of training pairs, some of the data are withheld from the system. Usually between ten and twenty percent of the available data are set aside to run through the trained network, testing the system's ability to correctly assess cases on which it has not trained. If the testing pairs are assessed with success similar to the training pairs, and if this performance is sufficient, the network is ready for actual use. If the testing pairs are not assessed with sufficient accuracy, the network parameters must be adjusted by the network designer, and the entire process is repeated until acceptable results are achieved. For some problems, this will never occur. Recent advances in neural network technology have significantly advanced the field, but no single AI paradigm can ever solve every problem.

The actual backpropagation algorithm fits within the framework described above. The theory and actual mathematical algorithm are quite involved and beyond the scope of this paper. The general concept is that the error adjustments backpropagated into the weight matrices during the reverse passes get smaller as the network gets more accurate. Thus if the problem is trainable, the system finally converges to an acceptably small total error.

Documented and intensely studied problems do exist with backpropagation (Wasserman, 1989). Network paralysis can occur when weights become too large due to excessive adjustment. When this occurs, the error which is backpropagated becomes very small, since the error is calculated as proportional to the derivative of the squashing function. At large summed neuron input levels, the slope (derivative) of the squashing function is near zero, and thus the backpropagated errors are near zero. Local minima sometimes cause the networks to stop training before an optimal set of synaptic weights is found. One way to view the backpropagation algorithm is traversing an error surface, trying to find the minimum point. However, since this algorithm only allows downhill travel, local minima sometimes are found, since these error surfaces are often convoluted. The algorithm cannot travel out of these local valleys, even if the true and much deeper minimum is very near. A third problem deals with the common computer tradeoff of accuracy versus runtime. One algorithm parameter is called step size, which is closely related to learning rate. If the step is too large, paralysis or instability results; too small and training time becomes very long.

One way in which backpropagation differs from human learning is that the same set of data must be cycled through the network over and over, until training is complete. Humans learn through a continuous stream of new data, but manage to

remember (most of) what has been previously learned. If this continuous, non-repeating data stream were fed to a backpropagating trainer, most things would be soon forgotten, even if learned perfectly initially.

While other neural net training methods and representations exist, none have successfully emerged as widely or commercially viable at this point. Several alternate technologies are very promising, though, and will be mentioned briefly here for completeness. Counterpropagation networks (Hecht-Nielsen, 1988) are basically look-up tables capable of generalization. They are very fast-running, but have limited applications. Statistical training methods (Geman, 1984) are useful for eliminating problems with local minima, but are very slow training. Hopfield networks (Hopfield, 1982), also called recurrent networks, train quickly and use computer space efficiently, but are unstable and have many other problems. Bidirectional associative memories (BAMs) (Kosko, 1987) are similar to Hopfield nets, but are modeled more after human learning. Problems include limited storage ability and unpredictable behavior. Adaptive resonance theory (ART) (Carpenter, 1987) suggests a method for learning without affecting previously learned data, but is very complex and currently still largely theoretical. Finally, cognitrons (Fukushima, 1975) are human-inspired systems that include elements of psychology and physiology. Though they show great promise, and model the human brain more closely than other systems, they are exceedingly complex and inefficient, requiring tremendous computing resources. Some of these methods are beginning to appear in commercial products, but backpropagation is still the most widely used algorithm by far. Given the breadth of ongoing neural net research, new standards are likely to emerge in the future.

In the description following Figure 4-2, the way in which neural nets are generally used was implied. Neural net shell programs provide facilities for the construction and exercising of user-trained networks. These shell programs are generally stand-alone applications, and cannot be embedded within other applications. However, some shell programs provide the option of creating source code outputs of the trained network that can be included in-line with other programs. Some programs allow the code to be generated in a variety of computer languages, such as FORTRAN, Pascal, or C. This code can then be accessed as a subroutine from within an existing program. This is possible because of the relatively simple mathematics involved in running trained networks. Recall from the example above that the required numeric manipulations consist mainly of matrix multiplication and squashing function evaluation.

Another fascinating way in which neural networks are sometimes used is through cascading systems. When used in this manner, the output neurons from one serve as input neurons in another system. This application of neural networks allows very complex objects to be modeled modularly, with subsystems or components each using their own individually-created neural networks. This type of application of most easily accomplished with source-code-level networks as described in the preceding paragraph. Using neural nets in this way is very consistent with human brain activity; neuroanatomists have determined that biological neurons often function in identifiable groups that send corporate signals to other neural groups.

4.2 POTENTIAL NEURAL NETWORKS FOR NUCLEAR SURVIVABILITY VALIDATION.

As with expert systems, there are several ways in which artificial neural networks could be used to solve existing problems with nuclear survivability validation. The issue of uncertainty was one that was evident on several levels. Uncertainty exists in

assessing threat characteristics. Nuclear environments need to be calculated for the effects listed in the problem statement, and such prediction is not an exact science. Finally, calculating the system response from the environments induces yet another level of uncertainty.

Neural nets, by their nature, have the ability to handle uncertain data. Most neural net outputs are estimates based on trends. The generalization that is the goal of almost all neural nets usually implies that the final result will be to correctly approximate the actual answer. This being the case, data which are not known with great precision can be modeled relatively easily.

Consider an example where one potential problem input is nuclear yield, and the yield is known only within twenty percent. In setting up the network, ten (for example) input neurons would be utilized for nuclear yield, each corresponding to a range of yields, probably scaled logarithmically. For training, where exact cases are known, the appropriate yield input neuron would be given a value of one, and the other yield input neurons would be set to zero. Once a valid system was in place and a run was to be made, the determination of what yield input neuron(s) would be activated could be calculated relatively easily. If the range of uncertainty falls entirely within one neuron's assigned range, that neuron would be set to one; all others to zero. If the uncertainty range spanned two (or more) neurons' assigned ranges, each would be given a fractional value corresponding to the probability that the actual value fell in its domain. Two adjacent yield neurons might have values of 0.7 and 0.3. The net would then be exercised with these (and possibly other) uncertain values to determine a "best guess" of whatever the desired output was (perhaps overpressure or EMP).

Similar "bin" neurons could be defined for any other uncertain quantities, in the threat, environment, effect, and response arenas. The number of bins used for each uncertain parameter would be a function of the required output precision and the general uncertainty inherent in the parameter. The concept described here is actually a combination of neural networks and fuzzy logic. Fuzzy set theory, which is based on the concept of partial set membership, will be discussed in detail in Section 5 of this document.

Another problem extant in performing nuclear survivability validation is integrating results from multiple simulations, multiple tests, or both, into a unified and consistent database. One of the problems that is seen in this area is that "bad" data will sometimes creep into the system. Neural nets have the ability to ignore clearly erroneous data, and have shown that in some cases upwards of ten percent bad data are tolerable.

When attempting to correlate data from different sources into a single resource, in many cases the sources will not agree. In these cases, a "credibility neuron" could be implemented, and codes that specialize, for example, in a certain regime of an effect could be given precedence in the appropriate cases. Again, fuzzy values could be used for this neuron. In cases where there is a vast and consistent difference in results, a neural net probably would not be able to train on the data. In such an event, a human decision would be required to determine the efficacy of the various tools or data.

A third area where neural nets seem a natural tool for NSV is that of multi-level system simulation integration. In modeling complex systems, or systems of systems, the natural paradigm for computer modeling would be to model subsystems (at all

levels) with modules, and have the executive program tie the modules together in a hierarchical network. Thus results from the lowest chosen level of simulation could be propagated to the higher layers in a way that would realistically emulate the actual underlying cause-and-effect relationships involved.

At the end of Section 4.1, the concept of cascading neural networks was introduced. Recall that to cascade neural networks, the output neurons of one neural net are fed directly into the input neurons of another net. Consider an example where one would like to model an incoming warhead. The lowest level neural networks would probably pertain to materials and piece-parts in the system. These networks would calculate materials responses to effects such as IR radiation, blast over-pressure, and so on, and would pass the effects on the materials up to the next level of integration, which would probably be the component level. The inputs for this network would be comprised of the output neurons of all applicable materials/piece-part networks. The outputs of each component network would then cascade to the next level, probably subsystems. This cascading would continue up to, and perhaps including, the system of systems level, if applicable. There would also likely be some cross-talk within the system; as the output of one material network may be needed in several component or subsystem nets. The final output neurons from the system, perhaps probability of kill or some other metric, would represent the propagation of many neural networks configured into several levels.

This concept could potentially be taken even further. The threat, response, and environment neural nets, along with the simulation/test integration networks, could be used as the lowest level networks in the above cascading system. The situation specifics could be fed into these networks, which would propagate their results to the materials/piece-parts level networks, and up from there. Without too much imagination, one could envision a single, large, neural net-based analysis tool. This tool would be maintainable due to extreme modularity, fast running (once trained, of course) due to the simple numerics involved, and would unify the approach to nuclear survivability validation in a way that has not been achieved before.

One final example of how neural nets could aid in NSV is in the area of survivability protocol and regimen development. For protocol development, a neural network could be constructed with input neurons corresponding to the subsystems and materials, and with output neurons for validation tools and methodologies. Binary inputs and outputs would be used here, with neurons used as toggle switches. A similar network could be envisioned for constructing regimens from protocols. These would be unusual applications of neural networks, but given the youth of the field, few applications could truly be considered common. At this point, there is no reason to believe that this approach would work better than an expert system, and in fact, this method of using neural nets emulates expert system operation.

As with the sample expert systems in Section 3.2, these potential applications listed here are examples of what could be done. This list is neither exhaustive nor exclusive. Those more familiar with the areas touched on above would likely be able to identify additional areas where tools such as these would prove useful.

4.3 NEURAL NETWORK ISSUES.

While artificial neural networks are the topic of intense research and development at this time, there are issues and problems with them that should be addressed before investing time or money into them. The widespread commercial application of neural net technology is a very recent occurrence. The breakthrough that ushered

in the current neural net interest is only about five years old. And as with expert systems in the 1980s, a large speculative bubble seems to be forming for neural nets. Whether this technology will gain mainstream acceptance is impossible to forecast at this time, but indications are that most agencies are having more success in implementing and deploying neural nets than has been had with expert systems.

One problem with neural networks is that they are not good at handling consistently inconsistent data. This is also true of traditional programs and expert systems, but it is especially troublesome here. In other types of programs, one usually would explicitly define relationships between the data, and one could easily test for contradictory data with hard-coded test statements. In expert systems, it is unlikely that contradictory rules would be derived, unless multiple experts were used. In any event, expert system tools are available to check for problems such as these.

Neural nets, however, have tremendous egos. They will happily train away on any data given them, without ever suspecting they could be doing something wrong. If the amount of contradictory data is relatively small, neural nets will smugly ignore the offending data and concentrate on what seems consistent. In cases where conflicting data are the rule rather than the exception, though, the network may well organize itself into an inapplicable system that handles the inconsistencies as a side effect. Worst of all, when the trained network is run, it will always confidently deliver an answer that has no foundation, without any hint that something is amiss.

This effect is magnified by the second main problem with neural nets: reasoning capacity. While traditional programs can be examined to determine their underlying logic, and expert systems will trace a rule path if requested, the reasoning behind a neural network's output is inaccessible. Recall that most of the knowledge lies in the hidden layers of the network. These layers are called hidden for a reason. After a network is trained, the sum total of its knowledge and abilities are found in a series of two-dimensional real-valued numeric matrices. A training algorithm determined these weight matrices with little help from the user. At this time, there is simply no way to determine why a neural net yields a certain answer.

Neural networks operate with astonishing speed. However, the preceding paragraphs have done little to inspire confidence that the systems do any more than calculate incorrect results with remarkable efficiency. The situation is not as grim as it may appear. The first problem underscores a need to have a reasonable understanding and confidence in the data that is to be used for training. This would be a prudent approach in any event. However, it is imperative that data is not simply thrown at the network for training without familiarity with the information. The self-training operation of neural networks does not excuse the system creator from understanding the problem domain at hand. Rather, it enhances the need for familiarity greatly.

As for the problems with reasoning opacity and the fact that a trained network will confidently answer any question, the solutions are a bit more subtle. First, as long as the data which was used for training is acceptable, severe problems should not occur within the range of data for which training was performed. If there is a question regarding accuracy within the range, additional test cases should be generated for the regime of interest. If necessary, portions of the database can be more densely sampled, and the network retrained. One must be careful, however, not to venture very far outside the domain of the training data. While neural networks have good generalization abilities, outside its realm of expertise this

generalization often fails. If these simple safeguards are followed, risk of error can usually be held to an acceptable level.

4.4 NEURAL NETWORKS VERSUS EXPERT SYSTEMS.

Artificial neural networks and expert systems are, in many senses, competing technologies. They compete for attention in the technical press. They compete for use by those in target industries. They compete for research grants and customer dollars. Individual institutions promoting one technology or another compete for prestige. Indeed, neural nets and expert systems seem to have been involved in a continual tug-of-war since the mid 1960s.

This is unfortunate, since in emulating human intelligence, the two technologies are very complementary. The quick pattern recognition and generalization provided by neural networks correspond to the mental functions that perform tasks requiring immediate response (Wasserman, 1989). Such tasks include balance, reflex, and instinct. Expert systems, conversely, model the logical, reasoning side of human intellect. This side of brain activity is generally used for performing less time-urgent tasks that require applied reasoning or deliberate action.

Thus expert systems are called for when a genius is needed, and neural nets are indicated when an idiot savant will do. Neural nets are good for performing repetitive tasks that would otherwise prove time-consuming. Expert systems are appropriate for limited applications that are difficult for the average person. It has already been noted that expert systems provide a much more evident reasoning trail than do neural networks. Another nice feature of expert systems is the ability to insert or modify specific knowledge to gauge the effect. When explicit knowledge is the main concern, expert systems excel.

On the other hand, one of the main features of neural networks is the ability to train a system without explicit knowledge. Generally, domain experts are not needed for neural network development, as the needed expertise lies implicit in the data used for training. If gauging the effect of additional information is necessary, retraining the network with the additional data is straightforward. Thus the main difference in constructing the two types of AI systems seems to be data collection versus knowledge engineering. The type of data that is available, as well as the type of task that is to be performed, should determine which technology is appropriate, if either.

SECTION 5

FUZZY SETS AND FUZZY LOGIC

The third and final technical concept described in this paper is fuzzy set theory and fuzzy logic. Fuzzy sets are a relatively recent development in the field of mathematics. While not developed as a subfield of AI, the concepts are very intertwined with both expert systems and neural networks. In addition, the popular press has seen fit to lump fuzzy theory in with AI. When considering that the definition of AI used in this paper involves the use of non-traditional methods to solve problems in a human-like way, fuzzy logic would appear to fit into the category nicely.

5.1 FUZZY LOGIC: TECHNICAL DISCUSSION.

The concept behind fuzzy logic is that of partial set membership. Traditional ("crisp") set theory states that any existing object is either in a particular set completely, or not at all. In 1965, Zadeh generalized crisp set theory by allowing objects to be partial members of a set. Set membership values range from zero to one, inclusive, with zero indicating nonmembership and one implying total membership. If only the extreme values are used, as in crisp sets, Zadeh's set manipulations work just as the traditional set operators do. Thus crisp set theory is a proper subset, or special case, of fuzzy set theory.

Fuzzy set syntax is similar to that of crisp sets. A crisp set definition might be:

$$A = \{w, x, y, z\}$$

where A is the name of the set, and w, x, y, and z are the members of A. A corresponding fuzzy set could be :

$$A = \{1.0/w, 0.6/x, 0.1/y, 0.75/z\}$$

where A is again the name of the set. Using this nomenclature, w is a complete member of A, x is 60 percent in A, and so on. The elements of fuzzy sets have two components each; an element and its degree of membership. The degree is listed first, and the two are separated with a slash.

Familiar crisp set operators have analogs in fuzzy theory. The union of two crisp sets is simply another set containing all elements of both sets. With fuzzy sets, the union is not quite as obvious. Consider the fuzzy sets A above and B as follows:

$$B = \{0.3/t, 0.9/u, 0.6/w, 0.8/x\}.$$

The definitions of fuzzy union and intersection are:

$$A \cup B = \{\max(a(x), b(x))/x \mid x \in U\} \text{ and} \\ A \cap B = \{\min(a(x), b(x))/x \mid x \in U\}$$

where A and B are fuzzy subsets of the universe U, and a(x) is the degree of membership of x in A and b(x) is the degree of membership of x in B (Schmucker, 1984). For the example sets above,

$$A \cup B = \{0.3/t, 0.9/u, 1.0/w, 0.8/x, 0.1/y, 0.75/z\} \text{ and}$$

$$A \cap B = \{0.6/w, 0.6/x\}$$

A more useful set would have an infinite number of elements and a membership criterion that is a function. For example, define the fuzzy set of youth as follows:

$$Y = \{m(x)/x \mid x \text{ is a positive real number}\},$$

and $m(x)$ is as shown in Figure 5-1.

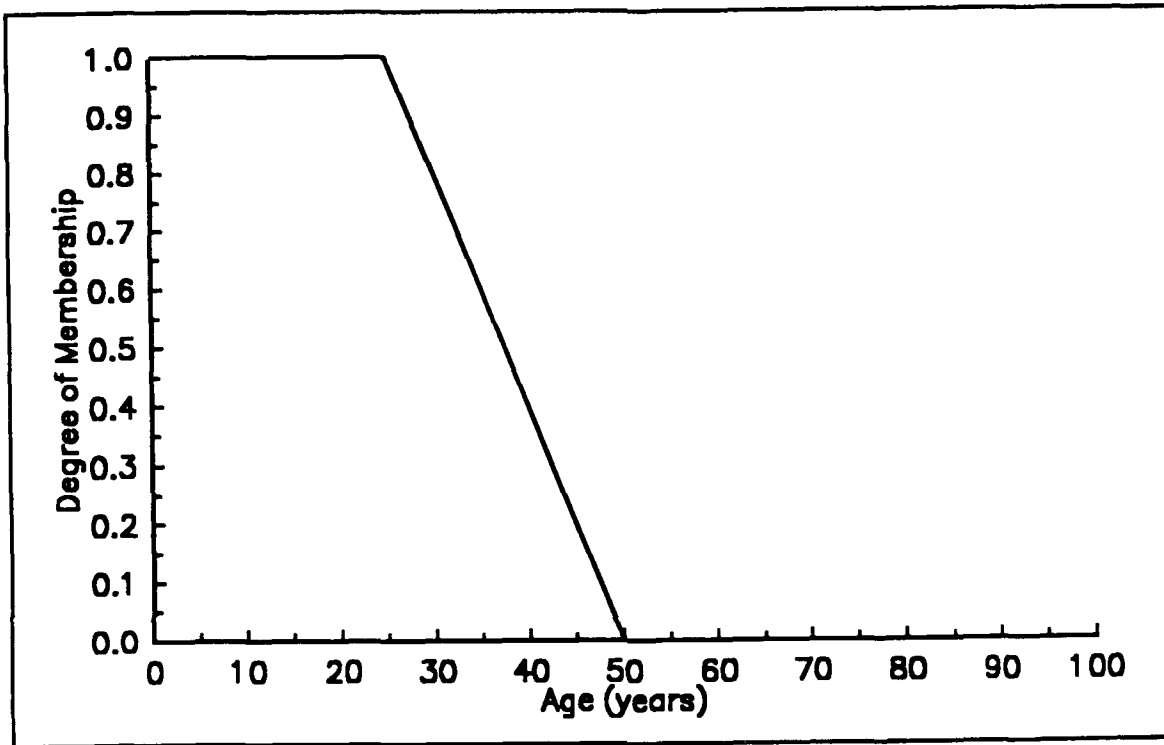


Figure 5-1. Youth fuzzy set membership.

The mathematical functions associated with Figure 5-1. are:

$$\begin{aligned} m(x) &= 1.0, \text{ for } x < 25; \\ m(x) &= -0.04x + 2, \text{ for } 25 \leq x \leq 50; \text{ and} \\ m(x) &= 0.0, \text{ for } x > 50. \end{aligned}$$

Using this fuzzy set definition, one could picture applications where age would be a determining factor, such as in defining a target marketing group that included such other fuzzy data as wealthiness and crisp data like education and geographical location. A targeted individual may be someone somewhat young, very wealthy, not living in New York state, and college-educated. Adjectives such as "somewhat," "very," and "not" are referred to as hedges, and are used to vary the shapes of fuzzy membership distributions. "Somewhat" is a dilating hedge, "very" is an concentrating hedge, and "not" is a reversing hedge. Other hedges are available to perform such transformations as normalization, intensification, and fuzzification.

5.2 POTENTIAL FUZZY LOGIC APPLICATIONS FOR NUCLEAR SURVIVABILITY VALIDATION.

It has been shown that the sole purpose of fuzzy logic is for dealing with uncertain data and concepts. Given this prime characteristic, there are several ways in which fuzzy sets and fuzzy logic could apply to NSV. In Section 1, uncertainties were shown to be a problem for accurately performing the needed tasks. Uncertainties were evident in threat characteristics, environment modeling, and systems response. In terms of fuzzy logic and AI, the most obvious applications would be embedded in expert systems or neural networks.

Expert systems have a simple form of uncertainty management included, in the form of confidence factors (see Section 3.1). This can be extended with the use of fuzzy logic. For instance, rather than creating an IF-THEN rule with numeric values in the antecedent, such as

IF warhead yield is greater than 1 megaton . . . ,

a rule could test for an imprecise concept

IF warhead yield is relatively large

A fuzzy membership function for warhead size would be created as shown above, and the corresponding outputs from this function could be used as confidence factors in both the antecedent and consequent of the rule. Thus if the U.S. were not able to ascertain the exact characteristics of a potential threat, it could still be addressed with approximate or vague data. Similar applications for environment and effects problems could be similarly constructed.

Neural networks have the ability to work well with fuzzy data with no conceptual extension whatsoever. There are two ways in which this can be accomplished. In the first one, an input neuron is defined for fuzzy set membership level in addition to the other applicable problem input neurons, and the value of the membership function is inserted along with the other input data. In some cases, neural networks have some problems training effectively on such data, and in these cases the second method often works. Here, one simply defines a set of neurons for the fuzzy concept being considered. Each neuron is assigned a range of fuzzy set membership values, and the applicable one (or ones, if an additional error range is used) are activated to a level greater than zero and less than or equal to one. Examples of both of these concepts were described in Section 4.2.

Fuzzy sets and fuzzy logic are used as embedded routines and data within other applications. While expert systems and neural networks are often (usually) used in a stand-alone modes, no such mode is applicable with fuzzy logic. Inclusion of fuzzy sets within other (non-AI) analysis codes in the NSV community is possible, though. Many of these physics and engineering codes use parameters such as warhead size or other uncertain concepts. There is no reason that fuzzy logic could not be incorporated into some of these codes to handle inherent uncertainties in a more rigorous fashion.

SECTION 6

SUMMARY/CONCLUSIONS

While nuclear survivability validation is a task that has been addressed at many levels for a number of years, problems do exist with current methodologies and tools. By definition of the problem domain, there is a great deal of uncertainty with which to deal. Uncertainties in threat definitions and scenarios are evident, as well as uncertainty in predicting threat environments and resultant effects. And uncertainties in integrating multi-test/multi-simulation data are common. System complexity continues to increase at all levels of integration as new technologies emerge and find their way into deployed systems. Thus the ability to assess these complex, highly-integrated devices in nuclear-degraded environments must keep pace with the incorporated technologies. Indeed, with multi-level interdependent systems now the rule, the ability to accurately gauge these synergistic effects becomes vital. As new requirements such as operate through and single event upset survival are defined, some effects that have in the past received a cursory treatment must now be thoroughly explored.

Artificial intelligence offers NSV researchers several tools to add to their existing inventory. In most cases, the AI tools will simply allow existing tools to be used more efficiently and effectively. In particular, the AI paradigms of expert systems, artificial neural networks, and fuzzy logic seem very applicable to the problems listed above.

Expert systems are indicated when modeling high-level human abilities or expertise is required. Examples of potential expert systems for integrating multi-level systems were presented, along with possible test data integration expert systems. NSV protocol and regimen generation was shown to be approachable with expert systems. Systems response categorization expert systems were described, as were expert systems for man-in-the-loop simulation modeling.

Neural networks show the most promise for the more numeric aspects of several of the problems described. Uncertainty handling is a natural application for neural net modeling, as hidden trends and generalities that might otherwise escape detection can often be found. The modeling of complex, integrated systems via cascading neural networks could well lead to a more consistent approach to NSV simulation than has ever been achievable. And such a cascaded neural net hierarchy, once trained, would allow extremely in-depth parametric studies, as neural networks operate with extreme speed.

Fuzzy set theory and fuzzy logic, while not truly subfields of AI, provide extensions to expert systems and neural networks that are required to address some of the problems above. In particular, fuzzy logic provides a powerful way to deal with uncertainty and imprecise data. Fuzzy parameters within both expert systems and neural networks would be straightforward to implement. Additionally, fuzzy logic might well be added to existing NSV tools to help handle necessary uncertainty and imprecision with better consistency.

AI presents a set of sound and mature technologies. A general rule (to which exceptions exist) is that AI works best when a fast, good answer is better than a slow, perfect one. Care must be used in the application of AI technologies, but no more than with any of the other methods used for NSV. Any computer tool that is trusted blindly or applied indiscriminately will tempt the fates, and AI is certainly no excep-

tion. But with a coordinated approach and a rigorous test program, AI holds promise to unlock doors for the NSV community that have previously proven impenetrable.

SECTION 7

REFERENCES

- Anderson, J., "A Simple Neural Network Generating an Interactive Memory," (UNCLASSIFIED) *Mathematical Biosciences* 14:197, 1972. (UNCLASSIFIED)
- Burr, D.J., "Experiments with a Connectionist Text Reader," (UNCLASSIFIED) *Proceedings of the First International Conference on Neural Networks*, 1987. (UNCLASSIFIED)
- Carpenter, G. and S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine," (UNCLASSIFIED) *Computer Vision, Graphics, and Image Processing* 37:53, 1987. (UNCLASSIFIED)
- Cottrell, G.W., P. Munro, and D. Zisper, "Image Compression by Backpropagation: An Example of Extensional Programming," (UNCLASSIFIED) *Advances in Cognitive Science* vol. 3, Norwood NJ: Ablex, 1987. (UNCLASSIFIED)
- Duda, R.O., J.G. Gaschnig, and P.E. Hart, "Model Design in the PROSPECTOR Consultant System for Mineral Exploration," (UNCLASSIFIED) *Expert Systems in the Micro-Electronic Age*, D. Michie, ed., Edinburgh: Edinburgh University Press, 1979. (UNCLASSIFIED)
- Feigenbaum, E.A., B.G. Buchanan, and J. Lederberg, "On Generality and Problem Solving: A Case Study using the DENDRAL program," (UNCLASSIFIED) *Machine Intelligence, Vol. 6*, B. Meltzer and D. Richie, eds., Edinburgh: Edinburgh University Press, 1971. (UNCLASSIFIED)
- Feigenbaum, E.A., "The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering," (UNCLASSIFIED) *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, 1977. (UNCLASSIFIED)
- Forgy, C., and J. McDermott, "OPS: A domain-independent production system language," (UNCLASSIFIED) *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, 1977. (UNCLASSIFIED)
- Fukushima, K., "Cognitron: A Self-Organizing Multilayered Neural Network," (UNCLASSIFIED) *Biological Cybernetics* 20:121, 1975. (UNCLASSIFIED)
- Geman, S. and D. Geman, "Stochastic Relaxations, Gibbs Distribution, and Bayesian Restoration of Images," (UNCLASSIFIED) *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6:721, 1984. (UNCLASSIFIED)
- Grossberg, S., "Contour Enhancement, Short-Term Memory, and Consistencies in Reverberating Neural Networks," (UNCLASSIFIED) *Studies in Applied Mathematics* 52:217, 1973. (UNCLASSIFIED)
- Hayes-Roth, F., D.A. Waterman, and D.B. Lenat, *Building Expert Systems*, Reading, MA: Addison Wesley, 1983, p. 9. (UNCLASSIFIED)
- Hebb, D., *The Organization of Behavior*, (UNCLASSIFIED) New York: Wiley, 1949. (UNCLASSIFIED)

Hecht-Neilsen, R., "Counterpropagation Networks," (UNCLASSIFIED) *Proceedings of the IEEE First International Conference on Neural Networks*, M. Caudill and C. Butler eds., San Diego: SOS Printing, 1988. (UNCLASSIFIED)

Hopfield, J.J., "Neural Networks and Physical Systems with Emergent Computational Abilities," (UNCLASSIFIED) *Proceedings of the National Academy of Science* 79:2554, 1982. (UNCLASSIFIED)

Kohonen, T., "Correlation Matrix Memories," (UNCLASSIFIED) *IEEE Transactions on Computers* 21:353, Piscataway NJ: IEEE, 1972. (UNCLASSIFIED)

Kosko, B., "Bidirectional Associative Memories," (UNCLASSIFIED) *IEEE Transactions on Systems, Man, and Cybernetics* 18(1):49, 1987. (UNCLASSIFIED)

Lawrence, J., *Introduction to Neural Networks: Computer Simulations of Biological Intelligence*, (UNCLASSIFIED) S. Luedeking, ed., Grass Valley CA: California Scientific Software, 1991. (UNCLASSIFIED)

Lenat, D., "Expert Systems: Part I - Fundamentals," (UNCLASSIFIED) *Proceedings of the Fourth National Conference on Artificial Intelligence*, 1984. (UNCLASSIFIED)

McCulloch, W.W. and W. Pitts, "A Logical Calculus of the Ideas Imminent in Nervous Activity," (UNCLASSIFIED) *Bulletin of Mathematical Physics* 5:115, 1943. (UNCLASSIFIED)

McDermott, J., RI: *A Rule-Based Configurer of Computer Systems*, (UNCLASSIFIED) Tech. Rept. CMU-CS-80-119, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, 1980. (UNCLASSIFIED)

Minsky, M.L. and S. Papert, *Perceptrons*, (UNCLASSIFIED) Cambridge, MA: MIT Press, 1969. (UNCLASSIFIED)

Newell, A., "Artificial Intelligence and the Concept of Mind," (UNCLASSIFIED) *Computer Models of Thought and Language*, R.C. Schank and K.M. Colby, eds., San Francisco: W.H. Freeman, 1973. (UNCLASSIFIED)

Parker, D.B., *Learning Logic*, (UNCLASSIFIED) Invention Report S81-64, File 1, Office of Technology Licensing, Stanford University, Stanford CA, 1982. (UNCLASSIFIED)

Pitts, W. and W.W. McCulloch, "How We Know Universals," (UNCLASSIFIED) *Bulletin of Mathematical Biophysics* 9:127, 1947. (UNCLASSIFIED)

Rosenblatt, F., *Principles of Neurodynamics*, (UNCLASSIFIED) New York: Spartan Books, 1962. (UNCLASSIFIED)

Rosenblatt, F., "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain," (UNCLASSIFIED) *Psychological Review* 65:386, 1958. (UNCLASSIFIED)

Rumelhart, D.E., G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," (UNCLASSIFIED) *Parallel Distributed Processing*, vol. 1, Cambridge: MIT Press, 1986. (UNCLASSIFIED)

Sejnowski, T.J., and C.R. Rosenberg, "Parallel Networks that Learn to Pronounce English Text," (UNCLASSIFIED) *Complex Systems* 3:145, 1987. (UNCLASSIFIED)

Slagle, J.R., *A Heuristic Program that Solves Symbolic Integration Problem in Freshman Calculus: Symbolic Automatic Integrator (SAINT)*, (UNCLASSIFIED) Ph.D. Diss. Rept. 5G-0001, Lincoln Laboratory, MIT, 1961. (UNCLASSIFIED)

Schmucker, K.J., *Fuzzy Sets, Natural Language Computations, and Risk Analysis*, (UNCLASSIFIED) Rockville, MD: Computer Science Press, 1984. (UNCLASSIFIED)

Shortliffe, E.H., *Computer-Based Medical Consultation: MYCIN*, (UNCLASSIFIED) New York: American Elsevier, 1976. (UNCLASSIFIED)

Tanzer, E. and G. Slutsker, "Why Fuzzy Logic is Good Business," (UNCLASSIFIED) *Forbes*, May 13, 1991. (UNCLASSIFIED)

van Melle, W., "A Domain-Independent Production-Rule System for Consultation Programs," (UNCLASSIFIED) *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, 1979. (UNCLASSIFIED)

Wasserman, P.D., *Neural Computing: Theory and Practice*, (UNCLASSIFIED) New York: Van Nostrand Reinhold, 1989. (UNCLASSIFIED)

Webros, P.J., *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, (UNCLASSIFIED) Masters thesis, Harvard University, 1974. (UNCLASSIFIED)

Weizenbaum, J., "ELIZA - A Computer Program for the Study of Natural Language Communication between Man and Machine," (UNCLASSIFIED) *Communications of the Association for Computing Machinery*, vol. 9, no. 1, January 1965. (UNCLASSIFIED)

Zadeh, L.A., "Fuzzy Sets," *Information and Control*, (UNCLASSIFIED) vol. 8, 1965. (UNCLASSIFIED)

APPENDIX A

AI HISTORY/BACKGROUND

The histories of the previously discussed subfields of AI present an interesting and useful set of data in themselves. Many lessons learned are included below, as well as milestones in AI research.

A.1 EXPERT SYSTEMS HISTORY/BACKGROUND.

Following the near abandonment of artificial neural networks in the early 1970s after the publication of *Perceptrons* (Minsky, 1969), expert systems received the lion's share of AI research funding. This was no accident; the primary author of *Perceptrons* (Prof. Marvin Minsky of MIT) was one of the most vocal early proponents of expert systems. Although there were a few well-known expert systems before 1969 (Slagle, 1961), expert systems caught on only after the downfall of neural networks. This is discussed in more detail below in Section A.2. One early application proved a spectacular success and caught the immediate attention of the technical press, paving the way for several other highly-touted early systems. That was the DENDRAL project (Feigenbaum, 1971) at Stanford, which considered nuclear magnetic resonance, mass spectrograph, and various other chemical experiment results to determine potential structures of an arbitrary unknown compound. DENDRAL's capabilities are considerable. To this day, it is acknowledged that "DENDRAL surpasses all humans at its task and, as a consequence, has caused a redefinition of the roles of humans and machines in chemical research" (Hayes-Roth, 1983).

After the publication of DENDRAL's success, a flood of expert systems emerged from laboratories worldwide. Among those pre-1980 expert systems are several that still rank as AI's most famous successes. One of the most well-known expert systems in existence is MYCIN (Lenat, 1984), which diagnoses and suggests treatments for infectious blood diseases. A panel of medical experts compared the efficacy of MYCIN directly with a group of physicians (including both blood disease experts and interns). MYCIN was judged equal or superior to all other participants. MYCIN was the first expert system to use a set of IF-THEN rules that was truly independent from other parts of the program, and it spawned several other broadly-known programs. EMYCIN (van Melle, 1979) (MYCIN without the IF-THEN rules) was used to create a cardiopulmonary function disease expert system called PUFF (Feigenbaum, 1977) and a mineral deposit locating program called PROSPECTOR (Duda, 1979).

Another early family of expert system applications originated at Carnegie-Mellon University, starting with PSG (Newell, 1973), a computer language for modeling human cognition. Over the course of several years, PSG evolved into the OPS (Forgy, 1977) series of languages which are still widely used, and ultimately into R1, an expert system still used for configuring DEC VAX computers (quipped the author on the name of the system: "I used to didn't know what an expert was, and now I R1.").

About 1980, expert systems began a migration from academic to commercial laboratories. Many large corporations created specialized internal expert systems to solve specific problems for which they currently had experts available. Simultaneously, a large number of start-up AI consulting and software companies sprang into existence. These firms offered their services to large corporations, identifying potential expert system applications and usually constructing the systems. Other AI

companies created expert system "shell" programs, allowing companies (or individuals) to create their own expert systems. Generally, the most successful applications from this period were created internally at large corporations with custom software.

The AI/expert system companies were largely a short-lived phenomenon, although several companies have survived with specialized or superior products/services. There were several reasons for this decline. First, many companies wanted to "jump on the AI bandwagon," using AI for AI's sake. In these cases expert systems technology was often not really necessary, and the results were, therefore, disappointing. When expert systems were created to perform mundane or simple tasks, it became clear to management that the return on the considerable investment was poor. The second reason for the decline of expert systems companies was that the early expert system shells were generally quite limited in their abilities. They tried to develop programs that would create expert systems for any application, and the result was that in their race to market, the products were actually not very powerful. Overhyped advertisements made many companies abandon AI after a few false starts. A third factor is that expert systems, contrary to advertisement, are not easy things to produce. The technical details behind this were discussed in Section 3.1.

The result of these problems is that expert systems are no longer held in the esteem that they once were. The vast majority of the start-up companies "went public" or were acquired by other firms shortly after their inception. Most of the time, the companies survived just long enough to make the founder a handsome profit. Had technical forces driven the evolution of expert systems rather than commercial forces, the situation would probably be much different today. Alternatively, the shakeout of the expert systems market may prove valuable in the long run. The companies and products that have survived are far superior to the rank-and-file shells that appeared in the early 1980s. Plus, with most of the hype gone, companies are using expert systems where they are actually needed, rather than creating them as showpieces for shareholders or customers. Many companies and agencies have used expert systems successfully over the past decade, and quietly continue to do so. Expert systems tools continue to evolve, and surviving expert system companies are again growing.

A.2 NEURAL NETWORK HISTORY/BACKGROUND.

Despite the recent blitz in the popular press, neural network research is perhaps the single most mature subfield of AI, at least in terms of years of study. Perhaps this is because it is not a "proper" subset of AI, rather it has emerged from the widely divergent fields of biology, psychology/learning theory, and computer science. The earliest pioneers in the neural network field in the 1940s had nothing to do with computer science, as digital computers were still in their infancy. Rather, they were neurobiologists, neuroanatomists, and learning theorists. The work of McCulloch (1943) and Pitts (1947) describing a mathematical model of neural activity and Donald Hebb's learning theories (1949) stand as landmarks about which most early neural network activity centered. This work and much that built upon it was modeled with electronic circuits in the late 1950s and on computers in the early 1960s.

With several fascinating applications in the early 1960s, the first AI boom began. Researchers produced single-layer neural networks for such previously unapproachable tasks as weather prediction, electrocardiogram analysis, and computer vision. Researchers also constructed the concept of multi-layer networks, but only knew

how to consistently train single-layer networks, which were referred to as perceptrons (Rosenblatt, 1958). While several neural nets made headlines, other researchers were having difficulty applying the technology to their problems. Some very simple and straightforward problem domains befuddled researchers for a number of years. In 1969, Prof. Marvin Minsky of MIT published *Perceptrons*, in which he described the reasons behind these shortcomings.

The main problem with perceptrons, it turned out, was one of linear separability. This was discussed in detail in Section 4.1, and suffice it to say here that most interesting problems are not linearly separable, and thus, showed Prof. Minsky, not modelable by perceptrons. Many such problems are solvable by multi-layer neural networks, but Prof. Minsky opined in his book that the problems with training multi-layer nets were insurmountable. He actually did urge research to investigate this further, but his stated opinion was sufficient to deal neural network research a near-fatal blow. Funding for neural network research all but disappeared, and public interest in AI waned until the expert system boom of the late 1970s.

The concerted effort of computer scientists, neurobiologists, and psychiatrists fractured at that point, and further neural net research was scattered among various academic institutions. A dedicated group of researchers kept neural network research active, and continued to make advances during the "dark ages" of neural nets. These individuals included James Anderson (1972), Teuvo Kohonen (1972), Steven Grossberg (1973), and Kunihiko Fukushima (1975). However, very little public attention was paid to neural networks until 1986, when Rumelhart, Hinton, and Williams derived an algorithm, called backpropagation, for training multi-layer neural networks. When their work became well-known, it was shown that the backpropagation algorithm had earlier been presented by Parker in 1982. Shortly thereafter, Webros was discovered to have described backpropagation even earlier, in 1974. The fact that three sets of researchers independently derived this technique over the course of twelve years speaks volumes on the fragmented state of neural net research of that period. Their collective research provided the breakthrough that was needed to make neural networks applicable to a much larger set of problems than was solvable by the simple perceptron.

Shortly after the backpropagation training method was developed and demonstrated, a great deal of activity occurred in neural net research and development. A flurry of papers was published detailing multilayer networks that solved problems that had proved intractable in the 1960s. Several commercial neural network shells came to market and were immediately applied to a much broader range of problems that could have been approached before. It is somehow appropriate that neural networks had a long dormant period, as computers in the 1960s would have been hard pressed to implement the new training algorithms. Training large neural networks via backpropagation is CPU, memory, and storage intensive.

Several famous neural nets were developed with backpropagation training techniques, and they have fueled the speculative fire much in the same way that the early expert systems did. The most famous of these, NetTalk by Sejnowski and Rosenberg (1987), converts text to phonemes (human language sound primitives), which can be easily routed to a speech synthesizer for conversion to sound. NetTalk proved better at pronouncing English words than any other algorithm yet produced. Exhaustive databases of human-derived pronunciation phonemes still exceed NetTalk's abilities, but NetTalk's ability to continue learning without human intervention, and with neither pronunciation rules nor a dictionary, is unique. Other

early famous multilayer networks performed such tasks as handwriting recognition (Burr, 1987) and image compression (Cottrell, 1987).

In addition to these early showpieces, a large number of networks have been constructed by individual agencies for their own use. California Scientific Software, a major neural network shell developer, reports (Lawrence, 1991) on a wide variety of networks that have been constructed using their and other software. These applications include:

NYU Medical Center:	Robot limb control
NASA:	Robotic object manipulation
General Dynamics:	Underwater sonar signal analysis
Univ. of PA/TRW:	Aircraft identification
Helsinki Univ. of Tech.:	Speech recognition
USAF:	Flight simulator support
General Devices:	Engine Problem Diagnosis
Eaton:	Multi-axle truck brake control
BC Hydro:	Power grid control
Halliburton:	Mudlog analysis
TWA:	Bomb detection

Backpropagation is not the ultimate advancement in neural network research. Rather, it provided a method that made neural nets practical for the first time. A great deal of research is ongoing in the neural net arena, though few new concepts have advanced very far beyond the laboratory at this point, as was discussed in Section 4.1. In addition to the software developments mentioned above, neural networks seem to have come full circle to the approach of the 1950s, when dedicated electronic circuits were devised for neural net simulation. Neural computing processing chips have been or are being developed by AT&T, CalTech, DARPA, Motorola, Nova Technology, Syntonic Systems, Intel, Micro Devices, Hitachi, and Bell Labs.

A.3 FUZZY THEORY HISTORY/BACKGROUND.

Compared to the rich and varied history of expert systems and neural networks, there is relatively little to be said regarding fuzzy logic history. Perceiving a need to be able to handle uncertainty in set theory better than was currently achievable, Russian-born Iranian computer scientist (working at Berkeley) Lofti A. Zadeh first described fuzzy theory in 1965. His work went largely unnoticed until the 1980s, when Japanese researchers applied his work to several problems. Commercial applications of fuzzy logic began appearing in Japan a few years ago, and the technology has become very common in many Japanese consumer products. Examples of fuzzy logic applications in Japan include automatic camera focusing, camcorder picture stabilization, washing machine automation, vacuum cleaners, air conditioners, rice cookers, microwave ovens, toasters, refrigerators, antilock brakes, and automatic transmissions (Tanzer, 1991). The U.S. has also jumped on the fuzzy bandwagon, with firms such as Eaton, GE, GM, HP, Honeywell, and Rockwell developing fuzzy commercial products.

AI researchers were among the first to apply the technology in the U.S. Fuzzy sets and the associated logic and set manipulations fit very nicely into both expert systems and neural network paradigms. This was described in detail in Section 5.2, but expert systems have always had a methodology related to fuzzy logic, and this has been extended. Neural nets work nicely with fuzzy neuron sets without any

special effort on the part of programmers or modified software for the analysis. Like neural networks, specialized chips have begun to appear to process fuzzy variables.

APPENDIX B

GLOSSARY

ABM - Anti Ballistic Missile
AGT - Aboveground Test
AI - Artificial Intelligence
ART - Adaptive Resonance Theory
AT&T - American Telephone and Telegraph
BAM - Bidirectional Associative Memory
BC - British Columbia
CalTech - California Institute of Technology
CPU - Central Processing Unit
DARPA - Defense Advanced Research Projects Agency
DEC VAX - Digital Equipment Corporation Virtual Address Extension
EMP - Electromagnetic Pulse
FORTRAN - Formula Translation
GE - General Electric
GM - General Motors
GPS - Global Positioning System
HP - Hewlett-Packard
IR - Infrared
LISP - List Processor
MCC - Microelectronics and Computer Technology Corporation
M.D. - Medical Doctor
MIT - Massachusetts Institute of Technology
NASA - National Aeronautics and Space Administration
NSV - Nuclear Survivability Validation
nudet - Nuclear Detonation
NYU - New York University
PA - Pennsylvania
ProLog - Programming with Logic
RLG - Ring Laser Gyro
SEU - Single Event Upset
TWA - Trans World Airlines
UGT - Underground Test
U.S. - United States
USAF - United States Air Force
XOR - Exclusive Or

DISTRIBUTION LIST

DNA-TR-92-82

DEPARTMENT OF DEFENSE

ARMED FORCES RADIOBIOLOGY RSCH INST
ATTN: TECHNICAL LIBRARY

ASSISTANT TO THE SECRETARY OF DEFENSE
ATTN: DR BARKER
ATTN: EXECUTIVE ASSISTANT
ATTN: LTCOL M CRAWFORD
ATTN: LTCOL MAGILL

DEF RSCH & ENGRG
ATTN: DIR NAVAL WARFARE & MOB
2 CYS ATTN: DIR TEST FACILITIES & RESOURCES
2 CYS ATTN: DIR WEAP SYS ASSMNT
2 CYS ATTN: DEP DIR TWP
ATTN: DEP DIR TEST EVAL
ATTN: DIR AIR WARFARE
2 CYS ATTN: DEP DIR R & AT
ATTN: DIR OFF & SPACE SYS
ATTN: DEP DIR S & TNF
2 CYS ATTN: DIR STRAT AERO & TN SYS

DEFENSE ADVANCED RSCH PROJ AGENCY
ATTN: ASST DIR ELECTRONIC SCIENCES DIV
ATTN: CHIEF SCIENTIST
ATTN: DEP DIR RESEARCH
ATTN: DIR AEROSPACE & STRATEGIC TECH
ATTN: DIR DEFENSE SCIENCES OFC
ATTN: DIR NUCLEAR MONITORING RESEARCH
ATTN: DIR PHOTOTYPE PLANNING OFC
ATTN: DIR TACTICAL TECHNOLOGY OFC
ATTN: DIR NAVAL TECHNOLOGY OFC
ATTN: DIRECTOR
ATTN: TTO

DEFENSE COMMUNICATION AGENCY
ATTN: JOHN SELISKAR

DEFENSE COMMUNICATION AGENCY
ATTN: ZENOBIJA PERRY

DEFENSE COMMUNICATIONS ENGINEER CENTER
ATTN: DOC CTRL DRTT
ATTN: DRTS
ATTN: R410

DEFENSE ELECTRONIC SUPPLY CENTER
ATTN: DESC-E
ATTN: DESC-EC
ATTN: MR JOHNSON DESC-SPP

DEFENSE INFORMATION SYSTEMS
ATTN: A200
ATTN: A330
ATTN: A400
ATTN: A500
ATTN: A700
ATTN: A800
ATTN: CODE A300 M DOOLY
ATTN: C4S/SCC
ATTN: H102 ASSOC DIR ENGRNG & TECH
ATTN: H110 ADV TECH OFC
ATTN: H396 TECH MGT CTR

DEFENSE INTELLIGENCE AGENCY

ATTN: D/GDIP
ATTN: DB-TPO
ATTN: DIW-1C P COLLINS
ATTN: DIW-4
ATTN: DT-1
ATTN: DT-1B
ATTN: DT-2A
ATTN: DT-4D ROY AMACHER
ATTN: DT J VORONA
ATTN: OGA-4A

DEFENSE LOGISTICS AGENCY

ATTN: DLA-F
ATTN: DLA-QE
ATTN: DLA-QES
ATTN: DLA-SCC
ATTN: DLA-SCT
ATTN: DLSMO

DEFENSE NUCLEAR AGENCY

2 CYS ATTN: ACCW
2 CYS ATTN: ACFE
3 CYS ATTN: ACST
ATTN: ACTL
ATTN: DDIR
ATTN: DFOP
ATTN: DFRA JOAN MA PIERRE
ATTN: DFTD
5 CYS ATTN: DIRECTOR
ATTN: MID
10 CYS ATTN: NANF
10 CYS ATTN: NASF
10 CYS ATTN: NSNS
10 CYS ATTN: NSSD
ATTN: OPAC
ATTN: OPNA
ATTN: OPNO
ATTN: OPNS
5 CYS ATTN: OTA
50 CYS ATTN: OTA CAPT D RICHLIN
ATTN: RAAE
ATTN: RAAE K SCHWARTZ
ATTN: RAEE
ATTN: RAEE AL CONSTANTINE
ATTN: RAEE CAPT R STIERWALT
ATTN: RAEE DR PALKUTI
10 CYS ATTN: RAEV
10 CYS ATTN: RARP
ATTN: SPDS
ATTN: SPSP
ATTN: SPWE
ATTN: SPWE
10 CYS ATTN: SPWE K PETERSEN
5 CYS ATTN: TDTR
2 CYS ATTN: TITL

DEFENSE TECHNICAL INFORMATION CENTER
ATTN: DTIC/FDAB

DEPARTMENT OF DEFENSE
ATTN: PEOCCU-UT COL L KARCH
ATTN: PMA-283-21

DNA-TR-92-82 (DL CONTINUED)

ATTN: NAVAL WARFARE & MOB
ATTN: DIR LAND WARFARE
ATTN: DEP DIR TEST EVAL

DEPUTY DIRECTOR OF DEF RSCH & ENGRG
ATTN: LTC P J ENGSTROM

DEPUTY UNDER SECRETARY OF DEFENSE
RESEARCH & ADVANCED TECHNOLOGY
ATTN: O LOMACKY

NATIONAL DEFENSE UNIVERSITY
ATTN: NDU-LD-CDC
ATTN: CLASSIFIED LIBRARY

NET ASSESSMENT
ATTN: DIRECTOR
ATTN: DOCUMENT CONTROL

OPERATIONAL TEST & EVALUATION
ATTN: DEP DIR OPER TEST & EVAL STRAT SYS
ATTN: SCIENCE ADVISOR
ATTN: DEP DIR RESOURCES & ADMIN

STRATEGIC DEFENSE INITIATIVE ORGANIZATION
ATTN: DA/DR GERRY

THE JOINT STAFF
ATTN: J-5
ATTN: J-6A J TOMA
ATTN: J-6E
ATTN: J-6F
ATTN: J-8 CAD S PATE
ATTN: J-8 EXECUTIVE OFFICE
ATTN: J-8 TSD
ATTN: J-8/NFAD J L GRUMBLES
ATTN: J8 NUCLEAR FORCE ANALYSIS DIV

DEPARTMENT OF THE ARMY

CHAPARRAL/FARR
ATTN: AMCPM-CF

DEFENSE SYSTEMS MANAGEMENT COLLEGE
ATTN: COL H E LINTON

HARRY DIAMOND LABORATORIES
ATTN: AMSCL-PA
ATTN: AMSLC-MI-FI M MARDEN
ATTN: DELHD-SE
ATTN: SLCHD-HPM
ATTN: SLCHD-NP-P K WARNER
ATTN: SLCHD-NW
ATTN: SLCHD-NW-E R L ATKINSON
ATTN: SLCHD-NW-EH S KHAN
ATTN: SLCHD-NW-ES
ATTN: SLCHD-NW-ES T BOCK
ATTN: SLCHD-NW-HPM H BRISKER
ATTN: SLCHD-NW-P
ATTN: SLCHD-NW-P A BEVEC
ATTN: SLCHD-NW-P M ABE
ATTN: SLCHD-NW-P J J CORRIGAN
ATTN: SLCHD-NW-PW T MAK
ATTN: SLCHD-NW-R POLIMADEI
ATTN: SLCHD-NW-RP
ATTN: SLCHD-NW-TN

ATTN: SLCHD-NW-TN R LINGEBACH
ATTN: SLCHD-NW-TS
ATTN: SLCHD-RI
ATTN: SLCHD-RP
ATTN: SLCHD-TL-WRF
ATTN: SLCHD-TN
ATTN: SLCSM-IM-TL
ATTN: SLCSM-AA D ROBERTS
ATTN: SLCSM-D COL J DOYLE

OFFICE OF ASSISTANT SECRETARY OF ARMY
ATTN: DAMA-PPM
ATTN: SARD-TR
ATTN: SARD-ZCS

U S ARMY BELVOIR RD&E CTR
ATTN: STRBE-FGM
ATTN: STRBE-FGP HUNGATE
ATTN: STRBE-H WEN H CHEN

U S ARMY CHEMICAL RSCH & DEV CTR
ATTN: SMCCR-DDP
ATTN: SMCCR-MSI
4 CYS ATTN: SMCCR-NB
ATTN: SMCCR-PPC
ATTN: SMCCR-PPS

U S ARMY ELECTRONIC RESEARCH & DEV CMD)
ATTN: PROJECT MANAGER

U S ARMY MATERIAL TECHNOLOGY LABORATORY
ATTN: COMMANDER
ATTN: DRXMR-HH
ATTN: SLCMT-BM
ATTN: SLCMT-OMM

U S ARMY MISSILE & SPACE INTELLIGENCE CENTER
ATTN: AIAMS-YRP BELCHER

U S ARMY MISSILE COMMAND
ATTN: AMCPM-CC-TM-SE
ATTN: AMCPM-CF-T
ATTN: AMCPM-HA-SE-MS
ATTN: AMCPM-HAER
ATTN: AMCPM-HD
ATTN: AMCPM-HD-G
ATTN: AMCPM-HDE
ATTN: AMCPM-ML
ATTN: AMCPM-RP-E
ATTN: AMSMI-RD-CS-R
ATTN: AMSMI-RD-ST-NB
ATTN: AMSMI-RD-TE-C-EM
ATTN: AMSMI-RD-TE-S
ATTN: PM/TO

U S ARMY STRATEGIC DEFENSE COMMAND
ATTN: CSSD-SA-EV
ATTN: CSSD-SA-EV R CROWSON
ATTN: CSSD-SL
ATTN: SFAE-SD-GST-E P BUHRMAN

U S ARMY TEST & EVALUATION COMMAND
ATTN: AMSTE-TA-F
ATTN: AMSTE-TA-F L TELETSKI

U S ARMY VULNERABILITY/LETHALITY
ATTN: AMSLC-BL-N BASSETT
ATTN: AMSLC-VL-NE DR J FEENEY

US ARMY COMMUNICATIONS-ELECTRONICS CMD
ATTN: PM SATELLITE COMMUNICATIONS

US ARMY ORDNANCE MISSILE & MUNITIONS
CTR & SCHOOL
ATTN: ATSK-CC BILL GREEN

USA SURVIVABILITY MANAGMENT OFFICE
ATTN: AMSLC-VL-NE DR J FEENEY
ATTN: F MANION
ATTN: SLCSM-SE J BRAND

DEPARTMENT OF THE NAVY

DEPARTMENT OF THE NAVY
ATTN: DIRECTOR
ATTN: JCM-04
ATTN: JCMG-707

GPS NAVSTAR JOINT PROGRAM OFFICE
ATTN: CHARLES TABBERT

MARINE CORPS R&D ACQUISITION COMMAND
ATTN: CODE PSA CAPT G MISLICK

MARINE CORPS RESEARCH, DEVELOPMENT
ATTN: CODE PSE-C S BELLORA

NAVAL POSTGRADUATE SCHOOL
ATTN: CODE 1424 LIBRARY
ATTN: PHYSICS DEPT PROF K WOHLER)
ATTN: PROF R BALL CODE AA/BP

NAVAL RESEARCH LABORATORY
ATTN: CODE 2627
ATTN: CODE 4600 D NAGEL
ATTN: CODE 4613 A B CAMPBELL
ATTN: CODE 6180
ATTN: CODE 6303 R GULARTE
ATTN: CODE 6550 M PAULI
ATTN: CODE 6800

NAVAL SPACE COMMAND
ATTN: CODE VN313 J TRAMMEL

NAVAL SURFACE WARFARE CENTER
ATTN: CODE G13 B STROTHER
ATTN: CODE H33 M POMPEII
ATTN: COMMANDER

NAVAL SURFACE WEAPONS CENTER
ATTN: MAJ M PLUMER

NAVAL WAR COLLEGE
ATTN: CODE E-111

NAVAL WARFARE ASSESSMENT CENTER
ATTN: CODE 12A
ATTN: CODE 34W
ATTN: DOCUMENT CONTROL

NAVAL WEAPONS CENTER
ATTN: T ANDERSON
ATTN: CODE 3181 B KOWALSKY
ATTN: CODE 3181 T BELL
ATTN: CODE 3181 DRIUSSI
ATTN: CODE 32
ATTN: CODE 3433 B BABCOCK
ATTN: CODE 39104
ATTN: CODE 3917

NAVAL WEAPONS EVALUATION FACILITY
ATTN: CODE 20 S MAUK
ATTN: CODE 22 RAY C TERRY
ATTN: CODE 224 A ALDERETE

NAVSEA
ATTN: J SATIN

OFFICE OF NAVAL RESEARCH
ATTN: CODE OCNF-10P5
ATTN: CODE OCNR-1114
ATTN: CODE OCNR-1114SE
ATTN: CODE OCNR-1114SP
ATTN: CODE OCNR-1114SS
ATTN: CODE OCNR-12
ATTN: CODE OCNR-12D
ATTN: CODE OCNR-121
ATTN: CODE ONCR-00F
ATTN: CODE ONCR-10E
ATTN: CODE ONCR-10P4
ATTN: CODE ONCR-10P6
ATTN: CODE ONCR-111
ATTN: CODE ONCR-111D
ATTN: CODE ONCR-1112A1
ATTN: CODE 1112
ATTN: CODE 1132SM

OPERATIONAL TEST & EVALUATION FORCE
ATTN: COMMANDER

SPACE & NAVAL WARFARE SYSTEMS CMD
ATTN: PD-40
ATTN: PMW-156-13A
ATTN: PMW-142
ATTN: PMW-145
ATTN: PMW-146
ATTN: PMW-147
ATTN: PMW-151
ATTN: PMW-152
ATTN: PMW-153
ATTN: PMW-180-21

DEPARTMENT OF THE AIR FORCE

AERONAUTICAL SYSTEMS DIVISION
ATTN: ASD/ENACE
ATTN: ASD/ENSSS
ATTN: ASD/RWWI

AF SPACE COMMAND
ATTN: LKNIP MAJ S HOFF
ATTN: SM-ALC DET 25 P C LARTER
ATTN: DOCE

DNA-TR-92-82 (DL CONTINUED)

AIR FORCE INSTITUTE OF TECHNOLOGY/EN
ATTN: LTCOL R TUTTLE

AIR FORCE MATERIAL COMMAND
ATTN: AFMC/XRNA CAPT K HUNIGAN
ATTN: YAGD MSGT R ROGAN

AIR FORCE OPERATIONAL TEST & EVAL CTR
ATTN: CNP
ATTN: DR M WILLIAMS
ATTN: LGAW
ATTN: LG4S
ATTN: MAJ ARTMAN
ATTN: MAJ MARC STRICKLAND
ATTN: OAS R DEFFENBAUGH
ATTN: OASZ
ATTN: OAW
ATTN: OAWF
ATTN: OAWV
ATTN: OAY
ATTN: OAYS
ATTN: ST
ATTN: TEFG
ATTN: TEKA
ATTN: TET
ATTN: TEWD
ATTN: TEZN MAJ TOM HERRING
ATTN: TZN

AIR FORCE SPACE COMMAND (LKI)
ATTN: CAPT WEIDNER
ATTN: LT D BARRON
ATTN: MAJ D ROBINSON
ATTN: STOP 7

AIR FORCE SPACE TECHNOLOGY CENTER
ATTN: DR J JANNI
ATTN: SWL V OTERA

FOREIGN AEROSPACE SCIENCE AND
ATTN: FASTC/TAIX CAPT S HANCOCK

FOREIGN TECHNOLOGY DIVISION
ATTN: G CYR
ATTN: LT G MASSE
ATTN: TTX/-1 CAPT S HANCOCK

PHILLIPS LABORATORY
ATTN: NTAAC
ATTN: NTES LTCOL T BRETZ
ATTN: OL-NS/EN
ATTN: PL/WS L CONTRERAS

ROME LABORATORY
ATTN: COMMANDER

SPACE DIVISION(AFSC)
ATTN: MHE
ATTN: SSD/MHC
ATTN: SSD/MS
ATTN: SSD/MSGA LT R MILLER
ATTN: SSD/MZS
ATTN: SSD/MZS
ATTN: SSD/MZSE

WRIGHT LABORATORY
ATTN: D WATTS

WRIGHT RESEARCH & DEVELOPMENT CENTER
ATTN: FIBR ANALYSIS & OPTIMIZATION BR
ATTN: AFWAL/FIAA J BYRNES
ATTN: WRDC/MLPJ C PELLERIN

DEPARTMENT OF ENERGY

DEPARTMENT OF ENERGY
ATTN: C MEYERS

LAWRENCE LIVERMORE NATIONAL LAB
ATTN: DIRECTOR
ATTN: L-10 W E FARLEY
ATTN: L-13 W DUNLOP
ATTN: L-153 M BLAND
ATTN: L-153 M ONG
ATTN: L-156 A POGGIO
ATTN: L-20 DR G MILLER
ATTN: L-24 F KOVAR
ATTN: L-262 PAUL GUDIENSEN
ATTN: L-302 W WHITESELL
ATTN: L-389 B WEINSTEIN
ATTN: L-389 F EBY
ATTN: L-389 G STAEHLE
ATTN: L-389 P EBERT
ATTN: L-389 T HARPER
ATTN: L-439 J KELLER
ATTN: L-491 R WIRTENSON
ATTN: L-54 J RONCHETTO
ATTN: L-6 L WOODRUFF
ATTN: L-610 F E FROST
ATTN: L-83 M GUSTAVSON
ATTN: L-83 DR P HERMAN
ATTN: L-84 G POMYKAL
ATTN: L-85 K JOHNSON
ATTN: L-85 P CHRZANOWSKI
ATTN: L-95 PAUL C WHEELER
ATTN: W CROWLEY

LOS ALAMOS NATIONAL LABORATORY
ATTN: A112
ATTN: B229 J BARAN
ATTN: G SCHROEDER
ATTN: J CARTER
ATTN: M GILLESPIE
ATTN: D STROTTMAN
ATTN: MS P364
ATTN: REPORT LIBRARY
ATTN: R BEHRENS
ATTN: M SCHICK
ATTN: R KIRBY
ATTN: MAJ WILLIS
ATTN: L DEYOUNG

SANDIA NATIONAL LABORATORIES
ATTN: DIV 5311 R GALLAGHER
ATTN: DIV 5371 D HENSON
ATTN: DIV 8242 M BIRNBAUM
ATTN: TECH LIB PERIODICALS

SANDIA NATIONAL LABORATORIES
ATTN: A CHABAI DIV 9311
ATTN: DEPT 5160

ATTN: DEPT 9350 RAD EFFECTS
 ATTN: DIV 1231 J MAENCHEN
 ATTN: DIV 5214 J S PHILLIPS
 ATTN: DIV 9341 PAUL RAGLIN
 ATTN: DR J WIRTH ORG 5800
 ATTN: F DEAN DIV 5153
 ATTN: L M CHOATE DIV 9340
 ATTN: L TROST
 ATTN: ORG 4061 J ARFMAN
 ATTN: ORG 7263 R CASE
 ATTN: ORG 9300 J E POWELL
 10 CYS ATTN: ORG 9340 W BEEZHOLD
 ATTN: R HAGENGROBER
 ATTN: R J LAWRENCE DIV 1541
 ATTN: R KING DIV 9127
 ATTN: REPORTS REF 3144
 ATTN: TECH LIB 3141
 ATTN: W BALLARD DIV 9341

U.S. DEPARTMENT OF ENERGY
 ATTN: OMA/DP-252 I WILLIAMS
 ATTN: OMA/DP-252 MAJ D WADE
 ATTN: OMA/DP-252 R GUNDERSON
 ATTN: OP 241 F O'SHAUGHNESSY

DEPARTMENT OF DEFENSE CONTRACTORS

AEROSPACE CORP
 ATTN: D BARNARD
 ATTN: H BLAES
 ATTN: LIBRARY ACQUISITION
 ATTN: M HOPKINS
 ATTN: M WATSON
 ATTN: E BROUILLETTE
 ATTN: S GYETVAY
 ATTN: W BARRY
 ATTN: F FINLAYSON
 ATTN: P BUCHMAN
 ATTN: SEED M4/934 J CHAI

ANALYTIC SERVICES, INC (ANSER)
 ATTN: DR J SEELIG
 ATTN: DR S CROWLEY
 ATTN: LIBRARY
 ATTN: MSL SYS DIV MCN

APPLIED RESEARCH ASSOCIATES
 ATTN: R FLORY
 ATTN: S SPERRY

APTEK, INC
 ATTN: T MEAGHER
 ATTN: VICE PRESIDENT-ENGINEERING

BALL AEROSPACE SYSTEMS DIVISION
 ATTN: D L MURATA-SEAWALT

BATTELLE EDGEWOOD OPERATIONS
 ATTN: CBIAC J MCNEELY

BATTELLE MEMORIAL INSTITUTE
 ATTN: V PUGLIELLI
 ATTN: TACTICAL TECH CTR

BDM INTERNATIONAL INC
 ATTN: B POWERS
 ATTN: BV-36
 ATTN: E DORCHAK
 ATTN: J RYBICKI
 ATTN: J STOCKTON
 ATTN: O DOERFLINGER
 ATTN: W COOPER
 ATTN: W SWEENEY

BDM INTERNATIONAL INC
 ATTN: B TORRES
 ATTN: D E NASH
 ATTN: D WUNSCH
 ATTN: M HESSHEIMER
 ATTN: R HUTCHINS
 ATTN: R KARASKIEWICZ
 ATTN: R STEPHANS
 ATTN: R-33 B PLUMMER
 ATTN: T LAMBARSKI
 ATTN: W CORDOVA

BOOZ-ALLEN & HAMILTON, INC
 ATTN: AP-1 H KOHN
 ATTN: B SMITH
 ATTN: C HALL
 ATTN: F SIMMONS
 ATTN: J KEE
 ATTN: K SCHAFER
 ATTN: L ALBRIGHT
 ATTN: M SAUNDERS
 ATTN: R BALESTRI
 ATTN: R LEZEC
 ATTN: R RAPIDS
 ATTN: S ENG
 ATTN: V DEPRENGER

BOOZ-ALLEN & HAMILTON, INC
 ATTN: C HAYSLIPP
 ATTN: H WEBSTER
 ATTN: J KEE
 ATTN: R RAPIDS
 ATTN: T HANSON

BOOZ/ALLEN & HAMILTON INC
 ATTN: J LEE
 ATTN: R SWISTAK

CALSPAN CORP
 ATTN: M HOLDEN

CHARLES STARK DRAPER LAB, INC
 ATTN: B HARRIS

COMPUTER SCIENCES CORP
 ATTN: R SCHEMEL
 ATTN: T RADEBAUGH

DELCO SYSTEMS OPERATIONS
 ATTN: C CAPPS

DYNAMICS RESEARCH CORP
 ATTN: R CUCE

DNA-TR-92-82 (DL CONTINUED)

E-SYSTEMS, INC ATTN: TECH INFO CTR	LOCKHEED AERONAUTICAL SYSTEMS ATTN: CENTRAL LIBRARY
E-SYSTEMS, INC ATTN: DR W DICKSON	LOGICON R & D ASSOCIATES ATTN: C MO ATTN: DR T A PUCIK ATTN: G IVY ATTN: G K SCHLEGEL ATTN: G MESSENGER ATTN: G SAFONOV ATTN: J KING ATTN: LIBRARY
GENERAL DYNAMICS-FORT WORTH DIVISION ATTN: MZ-5997	
HUDSON INSTITUTE, INC ATTN: LIBRARY	
JAYCOR ATTN: D WALTERS ATTN: DR B C PASSENHEIM ATTN: E WENAAS ATTN: W SEIDLER	LOGICON R & D ASSOCIATES ATTN: D CARLSON ATTN: DOCUMENT CONTROL ATTN: S WOODFORD
JAYCOR ATTN: CYRUS P KNOWLES ATTN: E IRISH ATTN: E WENAAS ATTN: G THEROUX ATTN: H DICKENSON	LOGICON R & D ASSOCIATES ATTN: MANAGER
JAYCOR ATTN: B DAVIS ATTN: W CREVIER	M I T LINCOLN LAB ATTN: V SFERRINO ATTN: C F WILSON ATTN: V MISELIS ATTN: R HALL
JAYCOR ATTN: R POLL	MARTIN MARIETTA CORP ATTN: DR S ZEIBERG
JAYCOR ATTN: P MLAKAR	MISSION RESEARCH CORP ATTN: A BOLT ATTN: DOCUMENT CONTROL ATTN: E B SAVAGE ATTN: E PETTUS ATTN: J HAWXHURST ATTN: TECH INFO CENTER ATTN: W CREVIER
JAYCOR ATTN: C THOMPSON	MITRE CORPORATION ATTN: LIBRARY
KAMAN SCIENCES CORP ATTN: D COYNE ATTN: L MENTE ATTN: LIBRARY ATTN: R RAAB	NICHOLS RESEARCH CORPORATION ATTN: G SHELBY ATTN: HENRY T SMITH ATTN: K DENT ATTN: LARS ERICSSON ATTN: R BYRN
KAMAN SCIENCES CORP ATTN: ASST SUPV SCTY ATTN: C EKLUND 2 CYS ATTN: D RODVOLD 2 CYS ATTN: R WILLIAMS 2 CYS ATTN: W GATELEY ATTN: J EAMON ATTN: J HARPER ATTN: J KEITH 2 CYS ATTN: P BOOK ATTN: P WELLS ATTN: R GREER 2 CYS ATTN: T A STRINGER ATTN: W YOUNG	NICHOLS RESEARCH CORPORATION ATTN: L GAROZZO
KAMAN SCIENCES CORP ATTN: DASAC ATTN: E CONRAD ATTN: R GUENTHER	PHYSICS INTERNATIONAL CO ATTN: LIBRARY
KAMAN SCIENCES CORPORATION ATTN: DASAC	S-CUBED ATTN: DR M MONTGOMERY ATTN: DR C DISMUKES ATTN: DR G GURTMAN 3 CYS ATTN: H FREIBERG ATTN: J KNIGHTEN ATTN: J M WILKENFELD ATTN: R LAFRENZ ATTN: R WILSON ATTN: WAISMAN

S-CUBED

ATTN: J ATKINSON
ATTN: J NORTHROP

SCI TECHNOLOGY, INC

ATTN: DOCUMENT CONTROL

SCIENCE APPLICATIONS INTL CORP

ATTN: TECHNICAL REPORT SYSTEM

SCIENCE APPLICATIONS INTL CORP

ATTN: TECH LIBRARY

SPARTA INC

ATTN: H M BERKOWITZ

SPIRE CORP

ATTN: R EVANS

SYSTEM PLANNING CORP

ATTN: P MOON

TELEDYNE BROWN ENGINEERING

ATTN: P SHELTON

WEIDLINGER ASSOCIATES, INC

ATTN: T DEEVY