JCRL 15012
APPLIED DECISION ANALYSIS, inc.
3000 Sand Hill Road, Menlo Park, CA 94025 (415) 854-7101

2

DEFINITION, DEVELOPMENT, AND DEMONSTRATION OF
ANALYTICAL PROCEDURES FOR THE
STRUCTURED ASSESSMENT APPROACH

Final Report
Tasks 10, 11, and 12
LLL P.O. 9897503

MASTER

April 15, 1979

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

DISSEMINATION STATEMENT

CONFIDENTIAL

CONTENTS

I. INTRODUCTION	1
A Brief Background to SAA Development	1
Overview of ADA's Current Contribution to SAA Development	2
II. ADA SUPPORT TO THE SAA FACILITY X ASSESSMENT	4
Background	4
Description of the Factoring Algorithm	6
Overview of the Level 2 and Level 3 Analysis for Facility X	8
III. REQUIREMENTS FOR A PROBABILITY ALGORITHM FOR LARGE EVENT SPACES	13
General Problem Description	13
Assumption of the Number of Paths and Number of Components in a Path	14
Assumption of the Degree of Independence in Event Probabilities	14
Assumptions About the Types of Sensitivity Analyses	15
Illustrative Example of Algorithm Operation	17
Programming Requirements	19
Storage Requirements	20
Computation Requirements	21
Programming Language	22

List of Figures

2-1	Computation Diagram for Level 2 Analysis	5
2-2	Description of the Factoring Algorithm	9
2-3	Steps in Level 2 and Level 3 Analysis	10

List of Tables

2-1	Levels 2 and 3 Results	11
-----	----------------------------------	----

I. INTRODUCTION

Applied Decision Analysis (ADA) has assisted in refining analytical procedures for the Structured Assessment Approach (SAA) developed by Lawrence Livermore Laboratories (LLL). Chapter I of this report provides the background to ADA's current contribution to LLL's SAA and concludes with an overview of that contribution. Chapters II and III summarize specific ADA development activities.

A Brief Background to SAA Development

LLL developed the SAA as a systematic procedure for assessing the Material Control and Accounting (MC&A) systems at facilities that contain special nuclear material (SNM). This staged assessment methodology relies on efficient algorithms to manipulate basic design information automatically into a logical expression that describes the performance of the MC&A systems. These algorithms have two functions:

- To structure design information and data into a logical representation of system performance
- To calculate system performance measures for regulatory evaluations.

Portions of the SAA are more fully developed than others. At each stage of the analysis system performance measures have been defined and algorithms have been developed to calculate them. More work still remains in developing a specification for a methodology that can handle larger and more complex systems.

In logically structuring design information and data, the central and most difficult development area is manipulating Boolean expressions. The initial structuring of logical expressions for system performance is fully developed. This algorithm allows system design drawings, component reliability data to be transformed into a canonical form of the Boolean expression.

The next step, reducing this Boolean expression to identify important events and to calculate the performance measures, will require a more efficient algorithm to be developed. In the final step, probabilistic system performance measures can be calculated without difficulty with a reduced form of the Boolean expression.

Overview of ADA's Current Contribution to SAA Development

ADA performed three tasks during the reporting period:

- We participated in the demonstration assessment for Facility X by performing the probabilistic calculations for Level 2 and Level 3 of the SAA.
- We established the requirements for an efficient, feasible algorithm to be used in evaluating system performance measures that involve the probability of detection.
- We collaborated with other LLL project team members to identify and define the analytical relationship between the SAA detailed assessment procedures and the Aggregated Systems Model (ASM).

Chapter II describes ADA's role in assisting LLL to apply the SAA to Facility X. LLL provided the output of a Level 1 analysis, a representative material control system description, and a representative component probability data base. ADA used this material as input for a Level 2 and Level 3 analysis. In preparation for this demonstration of the SAA evaluation procedures, ADA developed a factoring algorithm that calculates the probability of detection for moderately sized material control (MC) systems. Both the algorithm and the results from the Level 2 and Level 3 analyses are discussed in Chapter II.

Chapter III describes the requirements for an algorithm to calculate the probability of detection for a given type of adversary and the target

set. This specification was derived from considerations of (1) the size of the event space for large and complex systems, (2) the probabilistic dependence of the occurrence of these events on one another, and (3) the ability to perform sensitivity analysis and provide intermediate results describing failure scenarios.

II. ADA SUPPORT TO THE SAA FACILITY X ASSESSMENT

In completing ADA's portion of the Facility X Assessment using the SAA, we improved the probability algorithm. This algorithm is more efficient than the method described in ADA's previous report "Overview of the Structured Assessment Approach and Documentation of Algorithms to Compute the Probability of Adversary Detection," T.R. Rice and S.L. Derby (December 15, 1978). Chapter III outlines an algorithm even more sophisticated than the approach outlined here.

Background

The SAA was applied to Facility X to demonstrate how the method applies to a full-size facility. Before this assessment was undertaken, the SAA was applied only to small examples used for illustration. The current demonstration shows that the SAA provides a series of system performance measures that can be useful in the overall evaluation of a SNM facility.

The Facility X Assessment was divided into an analysis of the MC&A systems. Both LLL and ADA worked on the MC system. LLL generated the monitor target sets (MTS) from a Level 1 analysis, as well as design information describing both the detection signal transmission network and the utilities network. The LLL project team analyzed the accounting system.

Figure 2-1 diagrams the steps in the Level 2 analysis. The MTS represent the groups of MC system monitors an adversary encounters during an attempted diversion. They are connected to a response authority through a signal flow path. If the response authority receives an alarm, detection has occurred. (Whether or not this authority acts to intercept the adversary is not part of the analysis.) The elements in the signal flow path are interrelated through connections in the utility network that

Computation Diagram for Level 2 Analysis

Figure 2-1

provides electric power, maintenance, and service. Using unit models to combine the signal path with the utility connections, the complete Boolean expression for detection is derived in terms of the availability of MC components. In this expression, all dependencies among monitors in a MTS are explicitly modeled. This expression, combined with a probability data base for all system components, is used in a reliability algorithm to calculate the MC system probability of detection for each MTS. Level 3 sensitivity analysis uses the Boolean expression and changes in probability data to determine significant failure events within the MC system.

The signal flow structure that ties the monitors to the safeguards authority at Facility X was unavailable to the project team. Therefore, to prepare Level 2 and Level 3 analyses using the procedures in the SAA, ADA employed illustrative information prepared by LLL to describe a representative MC system design. This illustrative system design contained 49 MTS and a system description containing 138 components.

For the Level 2 analysis, ADA defined the Boolean expression for detection for each MTS that employs the given system design. The Boolean expression was derived automatically with algorithms ADA had previously developed. Using component failure and availability data, ADA calculated the probability of detection with these Boolean expressions.

ADA developed a new probability algorithm to accommodate the more than 50 paths in the problem. (The previously developed algorithms had been limited by cost and computation time to approximately 15 paths.) With the development of the new, more efficient algorithm, ADA was able to complete the Level 2 and Level 3 analyses without excessive computation costs.

Description of the Factoring Algorithm

The factoring algorithm that ADA developed for the Facility X Assessment rearranges a minimal path representation of the Boolean expression for the detection event into mutually exclusive terms. In this form, the probability of detection is reduced to the sum of the probabilities for each term. Each term is composed of a set of factors, individual component

probabilities, and a reduced path representation. By factoring, the path representation in each term is reduced so that no common components occur in the paths. The probability of the mutually exclusive term can be calculated by multiplying each factor probability with the probability of the path representation. Because the path representation contains no common components, this probability is also a simple calculation.

The following example illustrates the Boolean manipulation used to factor paths. Two Boolean operations are used:

$$A + A \cdot B = A \quad (\text{Logical reduction})$$

$$B = B \cdot A \oplus B \cdot \bar{A} \quad (\text{Logical expansion})$$

where, $\oplus$ represents a mutually exclusive "OR" and $\bar{A}$ represents NOT A.

Using these operations, we can factor a path representation of a detection event containing common components into a set of mutually exclusive terms containing paths without common components. Let the detection event be described by the following expression:

$$D = A \cdot C + A \cdot B + B \cdot C$$

By factoring out the event A and using the first Boolean operation, we define a set of mutually exclusive terms:

$$\begin{aligned} D &= A \cdot (C + B) + B \cdot C \\ &= A \cdot (C + B) + (A \oplus A) \cdot B \cdot C \\ &= A \cdot (C + B + B \cdot C) \oplus A \cdot B \cdot C \end{aligned}$$

Next, using the second Boolean operation of logical reduction, we simplify the reduced path representation:

$$D = A \cdot (C + B) + A \cdot B \cdot C$$

The probability of detection is then simply the probability of the two mutually exclusive terms being summed together:

$$P_D = p(D) = p(A) p(C + B) + p(\bar{A}) p(B) p(C)$$

Figure 2-2 lists the steps in the factoring algorithm, describing the procedure to generate the mutually exclusive terms. First, the component event that is to be the factor is selected. Two new mutually exclusive terms are then created, with each term containing the factor and a new reduced path. If the paths remaining in each new term have a common component, the algorithm repeats the factoring procedure until the paths in all terms no longer have common components. An expression of mutually exclusive terms results. The sum of the probabilities of these terms is the probability of detection.

Overview of the Level 2 and Level 3 Analysis for Facility X

Figure 2-3 outlines the computational steps in the Level 2 and Level 3 analysis. First, a prototype algorithm combines the signal transmission paths with the utility network interconnections to derive automatically the path representation of the Boolean expression for detection for each of the 49 monitor target sets. Next, the new factoring algorithm ADA developed for this application transforms the path representation into an equivalent mutually exclusive set of terms for the Boolean expression. This factoring algorithm was applied to the dominant MTS determined by an analysis of the MC system structure. This Level 2 analysis shows that the reliability exceeded 0.94 for all MTS. The Level 3 analysis for the dominant MTS was limited to a single failure analysis for each component in the system. The results showed that the reliability for all MTS is sensitive to the availability for three types of system components -- the external power transmission line, the internal power transmission lines, and the alarm stations that receive the detection signal.

Table 2-1 summarizes the numerical results from the Level 2 and Level 3 analyses. The loss of the external power transmission line prevents detection by any MTS. Loss of an internal power distribution line or an alarm station lowers the MC system reliability to less than 0.9 -- the performance criterion assumed for the Facility X assessment.

Description of the Factoring Algorithm

1. Select the most common component.

FIRST TERM

2. Compute the probability of the component working.
3. Eliminate the factored component from each path in which it occurs.
4. Reduce the factored paths (if possible).
5. Store both the probability of the factor and the reduced set of paths for further factoring if paths remain dependent (i.e., if they have common elements). If the paths are independent, calculate the probability.

SECOND TERM

6. Compute the probability of the component's not working.
7. Eliminate all paths from the original set that contain the factored component.
8. If the reduced set of paths contain common components, continue factoring (return to Step 2). If the paths are independent, calculate the probability.

PROBABILITY OF DETECTION: Sum of term probabilities

Figure 2-2

Figure 2-3

Levels 2 and 3 Results

<u>Component Unavailable</u>		<u>MTS 1</u>	<u>MTS 2</u>	<u>MTS 3</u>
	None	0.9421	0.9421	0.9421
Power line	95	0.0	0.0	0.0
Internal	{ 96	0.8928	0.8928	0.8926
power lines		0.8925	0.8929	0.8929
Alarm	{ 135	0.8929	0.8929	0.8929
stations		0.8929	0.8929	0.8929

Table 2-1

A key qualitative insight emerges from these results -- the performance of this MC system design against this assumed adversary type can be modeled by treating only the reliability in five components. In effect, all but five components can be assumed to work with certainty. This significant reduction in the complexity of the system performance model can be used to guide the modeling for further analysis.

Volume II of the LLL report "The Assessment of Facility X" by A.A. Parziale, I.J. Sacks, T.R. Rice, and S.L. Derby (January 8, 1979), presents the Level 2 and Level 3 analyses, including a detailed description of monitor target sets, signal transmission networks, utility connections, component unit models, and probability data base.

III. REQUIREMENTS FOR A PROBABILITY ALGORITHM FOR LARGE EVENT SPACES

We calculate the probability of detection for a MC system in two major steps:

1. Reachability Analysis

A Boolean expression is defined in terms of the availabilities for the system components that describe the detection event for a given adversary. The Boolean expression for the detection event is expressed in "path" form as the logical sum of the various sets of components that can detect an adversary. The paths are generally not mutually exclusive; in fact, most expressions will contain many components among the paths.

2. Probability Analysis

The calculation of the probability of detection uses the path representation produced by Step 1, with the probabilities associated with each component event employed to evaluate the reliability of the complete MC system.

This chapter defines the requirements for the algorithm for Step 2, Probability Analysis, and describes the input.

General Problem Description

The probability analysis has three steps:

- The path representation must be covered to an equivalent expression with mutually exclusive terms.
- The probability of each term must be evaluated.

- The set of probabilities for the terms must be summed to produce the probability of detection.

The algorithm that completes these steps has three general requirements:

- The algorithm implementation must be able to handle many events ($< 1,000$)
- The algorithm must be able to handle both independent and conditional probabilities
- Event probability sensitivity analyses must be performed.

These three requirements cannot be met by a practical algorithm unless we make assumptions that further define the structure of the Boolean expression.

Assumption of the Number of Paths and Number of Components in a Path

A path with n events can potentially create 2^n mutually exclusive terms. The path representation of the Boolean expression potentially contains 2^{n-1} terms of n factors, and would require $2^{n-1}-1$ additions and $(n-1)2^{n-1}$ multiplications. If the expression were to be factored to the greatest degree possible, it would still contain $3(2^{n-1})-2$ factors, and require $2^{n-1}-1$ additions and 2^n-2 multiplications. Clearly, for large n , even the highly factored expression would defy evaluation by any modern computer.

In simplifying the specification of an algorithm, although 2^n mutually exclusive event sequences are possible, we assume relatively few will be meaningful. Furthermore, the algorithm relies on each of these paths involving only a portion of the n possible events.

Assumption of the Degree of Independence in Event Probabilities

If the probability of each event is independent of the occurrence of each of the other n events, then n probability specifications will be

required -- one for each event. If the probability of each event is dependent of the occurrence of each of the other n events, then a total of $2^n - 1$ probability specifications will be required.

The algorithm contains the assumption that the total number of conditional events will be relatively small. Specifically, most event probabilities are assumed to be independent; moreover, it is assumed that when dependencies do occur, they will involve only a few other events.

Assumptions About the Types of Sensitivity Analyses

For a given set of paths, two types of sensitivity analyses may be performed. The first involves analyzing the sensitivity of the result of the probability expression to numerical changes in event probabilities. The second involves analyzing the sensitivity of the structure of the probability expression to changes in the probability specifications (e.g., conditioning the probability of an event on the occurrence of another event).

The algorithm contains the assumption that sensitivity analyses will be limited to the first type. Because the second type requires an entirely new analysis, based on a changed path representation, the system should be defined with sufficient generality to ensure that all sensitivities are the first type. This compilation process contains three phases: input, transformation, and output.

Inputs to the compilation process are path specifications and event probability specifications:

- Path specifications are the output of the reachability analysis described earlier. Each path specification consists of a set of ordered pairs identifying the events in the path and their associated states (i.e., event occurs/does not occur).
- An event probability specification contains:
 - An event identifier

- A symbolic name to be associated with this probability specification
- A set of ordered pairs specifying an event identifier and its associated state (i.e., event occurs/does not occur). This list specifies those event occurrences on which the probability is conditioned. If the set is empty, the event probability is independent.

The probability specification need not describe the entire event space, and individual specifications may overlap. If a required probability is obtained from more than one specification during the production of the probability expression, a warning is issued, and the probability associated with the description of greatest dimensionality is used. Similarly, if a probability is required that has no specification provided, a warning is issued, and an appropriate default specification is provided.

The transformation from the input representation of the Boolean expression to the set of mutually exclusive terms consists of the following steps:

- Step 1. Read event probability specifications
- Step 2. Read all paths and form a reduced set of mutually exclusive paths
- Step 3. Factor each path description into independent subpaths
- Step 4. Factor out common subpaths and reduce the remaining subpaths
- Step 5. Repeat the previous step until no further factorizations/reductions are possible
- Step 6. Produce the probability expression for each unique subpath

Step 7. Combine the probability expressions for all unique subpaths to form the highly factored expression that represents the composite probability of the event subspace represented by the input path descriptions.

The output of the compilation process is an arithmetic expression in symbolic form. The expression is produced by employing the symbolic names supplied with the event probability specifications and observes the syntactic conventions of a high-level computer language such as PL/I, FORTRAN, or ALGOL.

Illustrative Example of Algorithm Operation

The following example shows how the algorithm operates.

Any path in the Boolean representation involves one or more of five events: A, B, C, D, and E. In specifying probabilities and event sequences, the nonoccurrence of an event is denoted by a horizontal bar over the symbol for the event. Thus, the sequences $A\bar{B}$ would mean, "Event A occurs; event B does not occur."

In this example, the probabilities of A, C, and D are independent, whereas the probability of B is conditioned on A and the probability of E is conditioned on both C and D. Furthermore, two paths exist that will result in successful diversion of SNM. Both paths require event B. One requires an additional sequence that involve D and E; the other requires the sequence $\bar{D}$ and $\bar{E}$.

Probability Specifications:

<u>Event Name</u>	<u>Symbolic Probability Name</u>	<u>Events On Which The Probability Is Conditioned</u>
A	A1	
B	B1	A
B	B2	$\bar{A}$
C	C1	
D	D1	
E	E1	D
E	E2	$\bar{C}\bar{D}$
E	E3	$\bar{C}\bar{D}$

Path Descriptions:

Path

B D E

B $\bar{D}$ $\bar{E}$

Because the two paths differ in the actual specification of at least one event, they are in mutually exclusive form. An analysis of the probability specifications reveals that each path contains two independent subpaths. For the first path, these subpaths are B and DE; for the second, the subpaths are B and $\bar{D}\bar{E}$. Factoring yields the equivalent path sequence of subpath B followed by either subpath DE or $\bar{D}\bar{E}$.

Using the input probability data, we expand the factored subpath sequences to produce the following probability expression:

$$T = (A1*B1+(1-A1)*B2) * (D1*E1+(1-D1) * (C1*(1-E2) + (1-C1) * (1-E3)))$$

where, T represents the detection. (The syntax employed here is essentially compatible with PL/I, FORTRAN, and ALGOL).

We can evaluate the expression by including it in a simple program that first inputs the values for variables A1 through E3. We can perform sensitivity analyses efficiently by varying the values of the variables.

Programming Requirements

The requirements discussed here are estimates intended solely for planning purposes. The actual requirements of the algorithm outlined above will not be known until it has been designed in detail. Those requirements will depend in large part on the specific technique developed to represent and process large, although sparsely populated, event spaces efficiently.

The actual programming requirements for any given set of event probability and path specifications will depend on the complexity of the conditional probabilities used in the probability expression. For example, if the probability of event B is conditioned on the occurrence of event A and the event sequence consisting of B is to be evaluated, we must obtain the probability expressions for both the sequences AB and $\bar{A}B$.

The estimates made in this section depend on the following assumptions about the algorithm implementation:

- Matrices and/or tree structures needed to represent probability descriptions, sequence descriptions, or probability expression structure will be maintained as sparse structures (i.e., only those elements or nodes actually containing data will be represented).
- Available computer memory will be allocated as required to represent probability descriptions, sequence descriptions, and probability expression structure; thus, no fixed limit will exist for the number of events, paths, path lengths, or probability specifications. (Independent probabilities need not be specified.)
- Whenever possible, descriptions of event subspaces, rather than explicit structural representations, will be processed.
- The algorithm will not be designed to produce a fully optimized expression; rather, it will perform those optimizations that can be processed efficiently in the memory available.

For example, a large number of conditional probability specifications may prevent the factoring of independent subpaths and subsequent probability expression generation from being performed on all paths simultaneously. In this case, optimization would be limited to subsets of the original input path set.

- External storage will not be used to any large extent for the data structures required by the algorithm (except for the external paging requirements of computers employing virtual storage systems).

Storage Requirements

The algorithm's storage requirements will depend primarily on the number of event probability specifications, the number of event sequence specifications, and the average path length of the event sequences. In the following expression, the symbol K represents 1024 bytes of storage.

Total storage (in bytes) = 500K

- + 12*(the number of unique events appearing in either sequence or probability descriptions)
- + 12*(the total number of event specifications appearing in event sequence specifications; i.e., the sum of all event sequence path lengths)
- + 12*(the number of event sequence descriptions)
- + 16*(the number of event probability specifications)
- + 8*(the total number of event specifications appearing in event probability specifications).

As an example, a set of 500 paths with an average path length of 100 events, together with 100 conditional probability specifications containing an average of two events and involving a total of 1000 unique events, would require approximately 1120K bytes -- a value well within

the capacity of most large, modern computer systems employing virtual memory management.

Computation Requirements

Computational requirements depend on the efficiency of the algorithm, the representation of the data structures, the execution speed and real storage capacity of the computer, and the efficiency of the object code produced by the compiler of the language in which the probability compiler is written. In other words, providing a performance analysis in the absence of a detailed design for the algorithm must be quite subjective. However, an estimate of the order of magnitude of the computational requirements gives some feeling for the practicality of the algorithm.

As an example, consider Step 2 of the algorithm as outlined in the "Assumptions About the Types of Sensitivity Analyses" section. "Read all paths and form a reduced set of mutually exclusive paths." The computational requirements of this step are significant because the data structures representing the paths must be built and because every input path must be checked against every other input path for common components.

Constructing the data structure for a sparse matrix representation of the event sequences involves computation of the linear function of the sum of the path lengths of the event sequences. If we assume that 10,000 machine instructions are sufficient to allocate the required storage for each event specified in an event sequence and to enter it at the appropriate place in the matrix, a set of 500 event sequences with an average path length of 100 events would require the execution of 5×10^8 machine instructions. For a computer with an execution rate of 5 million instructions per second (5 MIPS), constructing the data structure requires 100 seconds of CPU time.

Comparing every input event sequence with every other event sequence involves a process whose computational requirements are a function of the

average path length and the number of pairs that can be formed from the total number of paths.

If we assume that 100 machine instructions are sufficient to compare one path with another, a set of 500 paths with an average path length of 100 events would require

$$100 * (100 * (500 * 499) \div 2)$$

or 1.25×10^9 instruction executions. For a 5 MIPS machine, 250 seconds of CPU time would be required to perform the path comparisons.

Another 50 seconds of CPU time to convert paths with common components to an equivalent representation in mutually exclusive form must be added to the two requirements derived above. Thus, a total of 400 CPU seconds is required for a 5 MIPS machine.

If we assume that the computational requirements for Step 2 of the algorithm represent a substantial portion of the processing requirements (e.g., greater than or equal to one-tenth), the example under consideration could be evaluated in approximately 1 hour of CPU time for a 5 MIPS machine. This requirement is quite reasonable for a problem of this magnitude, although it is probably larger than the actual requirements an efficient implementation will need.

Programming Language

The principal processing that will be performed by the probability compiler will involve the creation and nonnumerical manipulation of the large, complex data structures. To meet the performance objectives, the language in which this algorithm is programmed must accommodate these activities conveniently and efficiently.

Choosing a high-level language for the algorithm over a machine code has four advantages. First, the expression generation can be performed without regards to the architect of the computer on which it is evaluated. Second, the expression will be in a form that can be used in computer programs that perform a dynamic evaluation of a probability expression. Third,

the expression can be examined easily (and modified, if required) by analysts involved in the reliability evaluation. Fourth, the expression can be converted to a fault tree representation by simply replacing the symbolic probability names by the names of the events they represent.

Of the commonly available high-level language, only PL/I is well-suited to the task of compiler implementation language. Although programming in either FORTRAN or ALGOL would be possible, the lack of data structures other than arrays would result in inefficiency that increases development time and computational overhead.

NOTICE

"Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore Laboratory under contract number W-7405-ENG-48."

"This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately-owned rights."

Reference to a company or product name does not imply approval or recommendation of the product by the University of California or the U.S. Department of Energy to the exclusion of others that may be suitable.