Mission Planning Analysis using Decision Diagrams

Theodore W. Manikas
High Assurance Computing and Networking Labs
Department of Computer Science and Engineering
Southern Methodist University
Dallas, Texas, U.S.A.
manikas@lyle.smu.edu

Mitchell A. Thornton
High Assurance Computing and Networking Labs
Department of Computer Science and Engineering
Southern Methodist University
Dallas, Texas, U.S.A.
mitch@lyle.smu.edu

Frederick R. Chang
21CT, Inc.
Austin, Texas, U.S.A.
fchang@21ct.com

Abstract—**Many military and space operations are phased missions, which contain non-overlapping phases. One approach for assessing phased mission system reliability is to apply binary decision diagrams (BDD's). While the BDD is an efficient structure for probability analysis, it cannot accurately represent all aspects of complex systems and processes as it assumes all phases follow binary logic. It is of great interest to ascertain if a system or process is to be successful during the planning and design stage. The concept and assessment of partial success when a system or process is in a degraded state is desirable and an objective of the methods presented here. To address the limitation of current analysis methods, we introduce the application of multiple-valued logic models to phased mission system analysis. These models identify the various levels of system operations and yield more information about the overall system operating states.**

Keywords—decision diagram, mission analysis; fault tree analysis; reliability; probability analysis

I. INTRODUCTION

A phased mission system consists of consecutive non-overlapping stages, or phases, where various tasks are performed in each phase. Examples of phased missions include military aircraft or ship operations, and space vehicle missions. The phases are usually executed in sequential order, so that the success of a given phase often depends on the success of its preceding phase [1], [2]. Analyses of phased missions require determining the failure probability of each phase as well as the entire mission. However, for large systems, the analysis process can become computationally expensive [3], [4].

For critical systems such as phased missions, it is important to assess the probability of success in the event of subsystem failure. Complete mission failure constitutes a disaster whereas complete mission success can sometimes be achieved even while some subsystems have failed. A complete disaster or failure of a component does not necessarily result in mission failure. Disaster tolerance is a superset of the more established approaches commonly referred to as fault tolerance. Models for disaster tolerance differ from those for fault tolerance since they assume that failures can occur due to massive numbers of individual faults occurring simultaneously or in a rapidly cascading manner as well as single points of failure. Therefore, a disaster-tolerant system can withstand a catastrophic subsystem failure and still function with some degree of normality [5], [6]. A phased mission system that is disaster-tolerant should be able to complete most of its mission despite the presence of phase failures.

A common approach for failure analysis of phased mission systems is to use fault trees, which have the form of binary logic circuits but are not intended as circuits, rather as a structure for fault analysis [1], [3], [4]. Fig. 1a shows an example of a simple binary fault tree. For fault trees, a logic level 1 indicates system failure, while a logic level 0 indicates no system failure [7]. If component C fails, then the entire system fails. Also, the system fails if both components A and B fail. Most past approaches for mission analyses do not adequately account for the presence of subsystem failure or degradation while still allowing for overall mission success due to the use of binary decision processes such as the use of fault trees.

For large systems, fault trees are often converted to binary decision diagrams for more efficient fault analysis. Decision diagrams are rooted directed acyclic graphs (DAG) that can be used to represent large switching functions in an efficient manner [8]. Efficient software is readily available to manipulate BDDs and a variety of heuristics and strategies have been adapted for use with fault trees [9-11]. Fig. 1b shows the corresponding BDD for the fault tree of Fig. 1a. The internal nodes are the components (A,B,C), and the leaf nodes are the resultant logic values for $f(A,B,C)$. The edge weights are the logic values for the components.

Fault trees and BDD's have been applied to phased mission system analysis [1-4]. However, a limitation of fault trees is their binary structure, which means that system phase operations can only be modeled as fully operational or completely failed. However, there may be mission phases that are partially operational, which will not adversely affect the overall success of the mission. Modeling different operational modes other than just the binary case of failure or normal operation are critical in analyzing the disaster-tolerance of phased mission systems. To address these issues, we propose the application of multiple-valued logic decision diagrams

(MDD) [14] and apply graph algorithms that compute the probability of mission success while also modeling mission component degradation or failure. Our proposed approach differs from previous mission planning approaches in that, to the best of our knowledge, this is the first time that multiple-valued logic has been applied to this problem.

The motivation for this approach lies in the fact that computation of the probability of successful completion of a process or function of a complex system can be difficult to predict. Because many components are specified with non-parametric reliability data, the cumulative effect for the reliability of the overall system is generally predicted through extensive Monte Carlo simulations or other statistical methods. By using an intermediate switching logic model, other approaches based on finite decision analysis are possible.

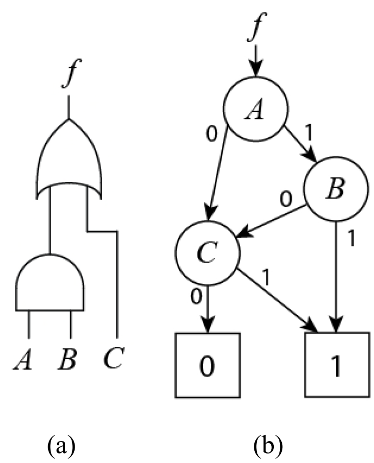

Fig. 1: (a) Fault Tree Example, (b) Corresponding BDD

II. MULTIPLE-VALUED LOGIC MODELS

Multiple-Valued Logic (MVL) systems are radix-p systems where $p>2$. For example, a radix-3 or ternary system is comprised of three logic values {0,1,2}. Therefore, the binary logical disjunctive (OR) operator generalizes to a *MAX* function, where $MAX(x,y,z)$ denotes the largest absolute logic value among the variables $\{x,y,z\}$. Similarly, the binary logical conjunctive (AND) operator generalizes to a *MIN* function, where $MIN(x,y,z)$ denotes the smallest absolute value among the variables $\{x,y,z\}$ [12]. Table I shows the truth table for the radix-3 function $f=MIN(A,B)$.

TABLE I. TRUTH TABLE FOR RADIX-3 MIN FUNCTION.

A	*B*	*f*
0	X	0
1	0	0
1	1	1
1	2	1
2	0	0
2	1	1
2	2	2

For MVL systems, an extension to the BDD construct has been developed and implemented called the Multiple-Valued Decision Diagram (MDD) [14]. Consider a totally-specified p-valued function with n inputs, $f(x_0,x_1,\ldots,x_n)$ where each dependent variable represents a value from the canonical logic set $x_i \in \{0,1,\ldots,p\text{-}1\}$. $f(x)$ is thus a finite discrete logic function and can be represented by the MDD data structure. Similar to BDD, the MDD is also a DAG and it consists of p terminal nodes, where each terminal node is annotated with a distinct logic value from the set $x_i \in \{0,1,\ldots,p\text{-}1\}$. Each non-terminal vertex is labeled with an input variable, and has a maximum of p outgoing edges, where each edge corresponds to each logic value. The MDDs used in this work are extensions from those described in [14] where the extensions allow for non-terminal vertices to have fewer than p outgoing edges and to allow for the representation of 'mixed-radix' vertices. Mixed radix vertices refer to collections of vertices corresponding to variables over the set $\{0,1,\ldots,q\}$ where $q<p$. Fig. 2 depicts a ternary MDD representing the function $f=MIN(A,B)$ corresponding to the truth table given in Table I.

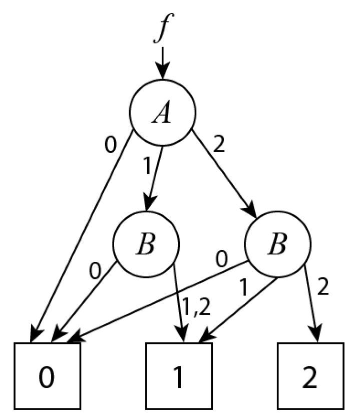

Fig 2: MDD for Example Radix-3 Function

A. *Application to Phased Mission Systems*

To illustrate the application to phased mission systems, we use the example given in [1] where the mission utilizes a UAV (Unmanned Aerial Vehicle) and a UCAV (Unmanned Combat Aerial Vehicle) that are cooperating to perform a military strike mission. The UAV mission has the following phases:

1. Launch – UAV is launched from its base.
2. Cruise1 – UAV cruises toward enemy target.
3. Surveillance – UAV performs surveillance of enemy target and transmits data back to base
4. Cruise2 – UAV returns to base.
5. Land – UAV lands at base.

The UCAV mission has the following phases:

1. Launch – UCAV is launched from base.
2. Cruise1 – UCAV cruises toward enemy target, based on surveillance information received from UAV.
3. Strike – UCAV strikes the enemy target.
4. Cruise2 – UCAV returns to base.
5. Land – UCAV lands at base.

The mission is modeled in [1] with the assumption that each mission phase can be classified as a binary outcome (1=failure, 0=success). The mission is considered to be a success if each phase is successful. Since the phases are sequential, if one phase fails, then the mission fails regardless of the subsequent phases. Thus, we can develop the truth table for the UAV mission as shown in Table II, where X = don't care. The approach of [1] uses BDD's to model the mission phases.

TABLE II. BINARY TRUTH TABLE FOR UAV MISSION SUCCESS.

Launch	Cruise1	Surveillance	Cruise2	Land	Mission Failure
1	X	X	X	X	1
0	1	X	X	X	1
0	0	1	X	X	1
0	0	0	1	X	1
0	0	0	0	1	1
0	0	0	0	0	0

Note the UAV mission is a success only if *all* mission phases have been successfully completed. However, this particular mission example utilizes the UCAV subsystem to perform its 'strike phase' if the UAV Surveillance phase is successful. Therefore, it is not necessary for the UAV to successfully complete its Cruise2 and Land phases for the UCAV to perform its phases.

If the UAV is able to complete its first three phases, then its mission is a "partial success". If the UAV is able to complete all its phases, then its mission is a "total success". Therefore, an alternate approach to the fault tree and corresponding BDD model is required to represent all possible phase levels of a mission and to allow for a more accurate analysis of the mission. Using a radix-3 MVL model, we let 2=total failure, 1=partial success, 0=total success. With these MVL assignments, we revise the truth table to represent phased mission model as follows:

1. For simplicity, assume that each phase of the UAV mission has logic level 0 or 2 (total success or failure). That is, each phase will have a binary outcome. However, the total mission success will have a ternary outcome as mentioned above.
2. If there is total failure during any of the first three phases, then the mission is a total failure.
3. If the first three phases are successful, then the mission is at least a partial success (logic level 1).
4. If all phases are successful, then the mission is a total success (logic level 0).

The revised truth table is shown in Table III, with the corresponding MDD depicted in Fig. 3.

TABLE III. RADIX-3 TRUTH TABLE FOR UAV MISSION SUCCESS.

Launch (*LN*)	Cruise1 (*C1*)	Surveillance (*SV*)	Cruise2 (*C2*)	Land (*LA*)	Mission (*f*)
2	X	X	X	X	2
X	2	X	X	X	2
X	X	2	X	X	2
0	0	0	2	X	1
0	0	0	0	2	1
0	0	0	0	0	0

Similarly for the UCAV, its mission would be a partial success if it were able to successfully complete its first three phases (Launch, Cruise1, Strike). Therefore, we can develop the UCAV mission truth table as shown in Table IV. The MDD will have the same structure as Fig. 3, but with the Strike node substituting for the Surveillance node.

TABLE IV. RADIX-3 TRUTH TABLE FOR UCAV MISSION SUCCESS.

Launch	Cruise1	Strike	Cruise2	Land	Mission
2	X	X	X	X	2
X	2	X	X	X	2
X	X	2	X	X	2
0	0	0	2	X	1
0	0	0	0	2	1
0	0	0	0	0	0

The success of the entire mission depends on the success of both the UAV and UCAV missions. If the UCAV is able to complete its Strike phase, this means that the UAV was able to complete its Surveillance phase. Therefore, we can develop the truth table for the entire mission as follows:

1. If either of the UAV or UCAV missions are total failures, then the entire mission is a total failure.
2. If both the UAV and UCAV missions are total successes, then the entire mission is a total success.
3. Otherwise, the entire mission is a partial success.

The truth table for the entire mission is shown in Table V, and its corresponding MDD is shown in Figure 4.

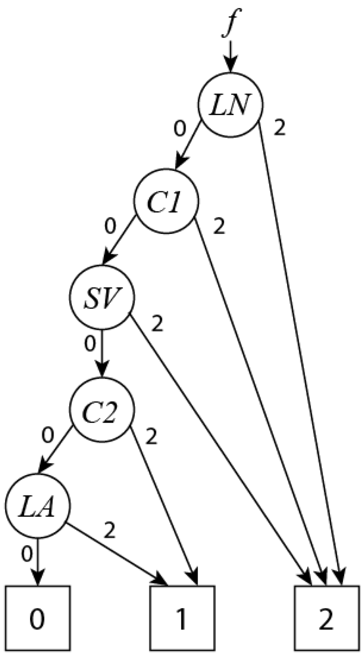

Fig. 3: MDD for UAV Success

TABLE V. RADIX-3 TRUTH TABLE FOR ENTIRE MISSION SUCCESS.

UAV	UCAV	Mission Failure
2	X	2
X	2	2
1	1	1
1	0	1
0	1	1
0	0	0

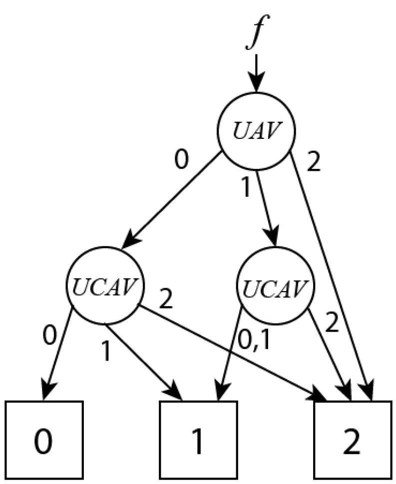

Fig. 4: MDD for Total Mission Success

B. *Probability Analysis*

The next step is to determine probabilities of mission success. Traditional probability theory assumes radix-2 systems: either an event *X* is true (X=1) or false (X=0). For general radix-*p* systems, we develop the notation X_j, which indicates that event *X* has state *j* ($j \in \{0,1,\ldots,p\text{-}1\}$). Thus, the probability that the UAV is at level 2 is represented by UAV_2, while the probability that the UCAV is at level 0 is represented by $UCAV_0$.

Assuming that the system probabilities are independent, we can calculate the UAV and UCAV probabilities by traversing the MDD's using the approach of [13]:

1. Represent phases as nodes A(Launch), B(Cruise1), C (Surveillance/Strike), D(Cruise2) and E(Land).
2. Recall that phase logic levels are either 2 (total failure) or 0 (total success).

The system probability equations become:

$$P\{\text{mission failure}\} = P\{2\} = P\{A2\} + P\{A0\}P\{B2\} + P\{A0\}P\{B0\}P\{C2\} \quad (1)$$

$$P\{\text{partial mission success}\} = P\{1\} = P\{A0\}P\{B0\}P\{C0\}P\{D2\} + P\{A0\}P\{B0\}P\{C0\}P\{D0\}P\{E2\} \quad (2)$$

$$P\{\text{total mission success}\} = P\{0\} = P\{A0\}P\{B0\}P\{C0\}P\{D0\}P\{E0\} \quad (3)$$

One of the advantages of this approach is that probability distributions may be non-parametric and based upon previously collected subsystem reliability data. Likewise, parametric distributions may also be used where each MDD variable is annotated with the appropriate probability distribution parameters. In this example, we utilize the non-parametric probability distribution supplied in [1]. Using these data from [1], we have the probability values for each mission phase as shown in Table VI.

TABLE VI. PROBABILITY VALUES FOR UAV/UCAV PHASES.

Phase	P{0}	P{2}
A	0.99	0.01
B	0.98	0.02
C	0.95	0.05
D	0.98	0.02
E	0.99	0.01

Using (1-3), we can calculate the individual mission probabilities of the UAV and UCAV (rounded to 4 significant digits) using Eq. 4-6. The probabilities correspond to MDD of Fig. 3.

$$P\{\text{UAV/UCAV mission failure}\} = P\{2\} = P\{A2\} + P\{A0\}P\{B2\} + P\{A0\}P\{B0\}P\{C2\} = 0.01 + (0.99)(0.02) + (0.99)(0.98)(0.05) = 0.07831 \quad (4)$$

$$P\{\text{UAV/UCAV partial mission success}\} = P\{1\} = P\{A0\}P\{B0\}P\{C0\}P\{D2\} + P\{A0\}P\{B0\}P\{C0\}P\{D0\}P\{E2\} = (0.99)(0.98)(0.95)(0.02) + (0.99)(0.98)(0.95)(0.98)(0.01) = 0.02747 \quad (5)$$

P{UAV/UCAV total mission success} = P{0} =
P{A0}P{B0}P{C0}P{D0}P{E0} =
(0.99)(0.98)(0.95)(0.98)(0.99) = 0.8942 (6)

We can also apply the approach of [13] to determine the probably of total mission success for the entire system (also rounded to 4 significant digits) using Eq. (7-9). We use the individual UAV/UCAV mission probabilities determined from Eq. 4-6 to determine the overall mission probabilities. These probabilities correspond to the MDD in Fig. 4.

P{mission failure} = P{2} = P{UAV2} +
P{UAV1}P{UCAV2} + P{UAV0}
P{UCAV2} = 0.07831 +
(0.02747)(0.07831) + (0.8942)(0.07831) =
0.1505 (7)

P{partial mission success} = P{1} =
P{UAV0}P{UCAV1} +
P{UAV1}P{UCAV0} +
P{UAV1}P{UCAV1} =
(0.8942)(0.02747) + (0.02747)(0.8942) +
(0.02747)(0.02747) = 0.04988 (8)

P{total mission success} = P{0} =
P{UAV0}P{UCAV0} = (0.8942)(0.8942)
= 0.7996 (9)

For our example phased mission system, we can see that the probability of at least a partial mission success is P{1}+P{0}=0.8495.

III. Conclusion

Many decision analysis problems are based upon non-parametric reliability information that is empirically collected. When complex processes or systems are analyzed using components with individual reliability data, the overall success or reliability of the system can be difficult to model.

Using the example of mission planning analysis, we have shown how such calculations can be accomplished automatically and efficiently using the concepts of multiple-valued logic and decision diagram data structures. We assign levels of subsystem degradation to each system component or subprocess allowing for MDDs to model either parametric or non-parametric reliability distributions. The use of this model then allows overall system success probabilities to be easily computed through directed graph traversal algorithms.

In future work, we intend to utilize the MDD-based model as a basis for the application of formal reasoning approaches whereby various applicable safety, liveness, and security properties can be formed using an appropriate logic framework and then formally proven to hold or be violated. The use of formal methods has potential for this application since the reliability model for a complex system or process is represented as a discrete logic function.

References

[1] J. D. Andrews, R. Remenyte-Prescott, and D. R. Prescott, "Reliability analysis of missions with cooperating platforms," in *Reliability and Maintainability Symposium* (RAMS)*, 2010 Proceedings - Annual*, 2010, pp. 1–6.

[2] D. Astapenko and L. M. Bartlett, "System design optimisation involving phased missions," *International Journal of Reliability and Safety*, vol. 3, no. 4, pp. 331–344, 2009.

[3] J. D. Andrews, D. R. Prescott, and R. Remenyte-Prescott, "A systems reliability approach to decision making in autonomous multi-platform systems operating a phased mission," *Reliability and Maintainability Symposium, 2008.* (RAMS). *Annual*, pp. 8–14, Jan. 2008.

[4] S. Reed, J. D. Andrews, and S. J. Dunnett, "Improved Efficiency in the Analysis of Phased Mission Systems With Multiple Failure Mode Components," *Reliability, IEEE Transactions on*, vol. 60, no. 1, pp. 70–79, Mar. 2011.

[5] S. A. Szygenda and M. A. Thornton, "Disaster Tolerant Computing and Communications," in Proc. Int. Conf. on Cybernetics and Information Technologies, Systems and Applications (CITSA), and Int. Conf. on Information Systems Analysis and Synthesis (ISAS), 2005, pp. 171–173.

[6] M. A. Harper, C. M. Lawler, and M. A. Thornton, "IT Application Downtime, Executive Visibility and Disaster Tolerant Computing," in *Proc. Int. Conf. on Cybernetics and Information Technologies, Systems and Applications* (CITSA)*, and Int. Conf. on Information Systems Analysis and Synthesis* (ISAS), 2005, pp. 165–170.

[7] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasi, **Fault tree handbook**, U.S. Nuclear Regulatory Commission, NUREG-0492, Jan. 1981.

[8] R. E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation," *Computers, IEEE Transactions on*, vol. C-35, no. 8, pp. 677 –691, Aug. 1986.

[9] L. M. Bartlett and J. D. Andrews, "Choosing a heuristic for the 'fault tree to binary decision diagram' conversion, using neural networks," *Reliability, IEEE Transactions on*, vol. 51, no. 3, pp. 344 – 349, Sep. 2002.

[10] K. A. Reay and J. D. Andrews, "A fault tree analysis strategy using binary decision diagrams," *Reliability Engineering and System Safety*, vol. 78, no. 1, pp. 45–56, 2002.

[11] A. B. Rauzy, J. Gauthier, and X. Leduc, "Assessment of large automatically generated fault trees by means of binary decision diagrams," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 221, no. 2, pp. 95 –105, Jun. 2007.

[12] D. M. Miller and M. A. Thornton, **Multiple Valued Logic: Concepts and Representations**. Morgan & Claypool Publishers, 2007.

[13] T. W. Manikas, M. A. Thornton, and D. Y. Feinstein, "Using Multiple-Valued Logic Decision Diagrams to Model System Threat Probabilities," in *Multiple-Valued Logic* (ISMVL)*, 2011 41st IEEE International Symposium on*, 2011, pp. 263–267.

[14] D.M. Miller and R. Drechsler, Implementing a Multiple-Valued Decision Diagram Package, in *Proc. IEEE Int. Symp. on Mulitple-Valued Logic* (ISMVL), 1998.