Reliability Evaluation of Multi-state Systems Subject to Imperfect Coverage using OBDD

Yung-Ruei Chang[1], Suprasad V. Amari[2], and Sy-Yen Kuo[1]

[1]*Department of Electrical Engineering*
National Taiwan University
Taipei, Taiwan
sykuo@cc.ee.ntu.edu.tw

[2]*Relex Software Corporation*
540 Pellis Road, Greensburg,
PA 15601, USA
suprasad.amari@relexsoftware.com

Abstract

This paper presents an efficient approach based on OBDD for the reliability analysis of a multi-state system subject to imperfect fault-coverage with combinatorial performance requirements. Since there exist dependencies between combinatorial performance requirements, we apply the Multi-state Dependency Operation (MDO) of OBDD to deal with these dependencies in a multi-state system. In addition, this OBDD-based approach is combined with the conditional probability methods to find solutions for the multi-state imperfect coverage models. Using conditional probabilities, we can also apply this method for modular structures. The main advantage of this algorithm is that it will take computational time that is equivalent to the same problem without assuming imperfect coverage (i.e. with perfect coverage). This algorithm is very important for complex systems such as fault-tolerant computer systems, since it can obtain the complete results quickly and accurately even when there exist a number of dependencies such as shared loads (reconfiguration), degradation and common-cause failures.

1. Introduction

The *s*-coherent multi-state system theory has been investigated since 1975 [1]. Many researchers have analyzed the *s*-coherent multi-state system reliability [2][3][4][5]. Most of them extend the concepts and conclusions for the 2-state *s*-coherent systems to the multi-state systems. To describe the dynamic characteristics of the component state transition, Stochastic process (Markov process) techniques are combined with the *s*-coherent multi-state system theory to analyze the dynamic multi-state system reliability. The multi-state reliability theory can handle situations in which the system and its components have a range of performance levels, e.g. from perfect operation to complete failure. Because performance degradation is very common in industrial products, it is important to develop the multi-state system reliability theory.

When a multi-state system (MSS) is considered, it is important to estimate the impact of each element on the system output/performance. The general definition of MSS reliability [2] is:

$$R_{MSS}(t,L) = \Pr\{F(t) \geq L\} \tag{1}$$

where L is the required performance level for MSS, $F(t)$ is the MSS output/performance rate. For a multi-state system that has a finite number of states, there can be H different levels of output/performance at time t:

$$F(t) \in \mathbf{F} = \{F_h, 1 \leq h \leq H\}$$

and the system output/performance distribution can be defined by two finite vectors **F** and

$$\mathbf{q} = \{q_h(t)\} = \Pr\{F(t) = F_h\}, \quad (1 \leq h \leq H)$$

Therefore, the non-repairable MSS reliability is the probability that the system remains in the states with $F_h \geq L$ during (0, t):

$$R_{MSS}(t,L) = \sum_{F_h \geq L} q_h(t) \tag{2}$$

In addition, systems that are used in life-critical applications such as flight control, nuclear power plant monitoring, space missions, etc., are designed with sufficient redundancy to be tolerant of errors. However, if the system cannot adequately detect, locate and recover from faults & errors in the system, then system failure can still result even when there exists adequate redundancy [6]. An accurate analysis must account for not only the complex system structure, but also the system fault and error recovery behavior. Therefore, the fault coverage problem of a system should be considered. This helps in fixing the optimal level of redundancy [7].

Most of published works use Markov models (non-homogenous Markov or semi-Markov model) to solve multi-state problems [8]. However, it is difficult to find the correct model of a system and there will be a total of $N = (m+1)^n$ states if there are n modules in the system and each module has (m+1) states including the imperfect coverage state. The computational time is proportional to $N^3 = [(m+1)^n]^3$. Hence, the computational complexity of the problem is $O(m^{3n})$. It is not just an NP problem, there are NP sub-problems within each step of the NP problem. This

Proceedings of the 2002 Pacific Rim International Symposium on Dependable Computing (PRDC'02)
0-7695-1852-4/02 $17.00 © 2002 **IEEE**

paper provides a new approach to model a multi-state system and proposes an efficient method combined with conditional probability concepts and OBDD method to evaluate the reliability of a multi-state system with imperfect coverage. This method could also be extended to use modularization methods for reliability analysis. This efficient integration of OBDD and modularization method simplifies the problem further.

Section 2 introduces the concepts of OBDD and coverage model. Section 3 illustrates a new model and a new approach to evaluate the reliability of a multi-state system with imperfect coverage. Section 4 proposes an OBDD-based algorithm to deal with the dependency problem in the probability evaluation of a multi-state system with imperfect coverage. Section 5 gives some examples. The last section gives the conclusions and future works.

2. Preliminaries

2.1. Ordered Binary Decision Diagram (OBDD)

This section introduces the representation and manipulation of Boolean functions based on OBDD. OBDD [9] is based on a decomposition of Boolean function called the *Shannon expansion*. A function f can be decomposed in terms of a variable x as:

$$f = x \cdot f_{x=1} + \bar{x} \cdot f_{x=0}$$

A node and its descendants in an OBDD represent a Boolean function f, where for node label x, one outgoing edge is directed to the subgraph representing $f_{x=1}$, and the other to $f_{x=0}$. Shannon decomposition is the basis for using OBDD. In order to express Shannon decomposition concisely, the if-then-else (*ite*) format [10][11] is defined as:

$$f = ite(x, f_{x=1}, f_{x=0})$$

2.2. Manipulation of OBDD

The manipulation of OBDD to represent logical operations is simple. In practice, the OBDD is generated by using logical operations on variables. Let Boolean expressions f and g be:

$$f = ite(x, f_{x=1}, f_{x=0}) = ite(x, F_1, F_0)$$
$$g = ite(y, g_{y=1}, g_{y=0}) = ite(y, G_1, G_0)$$

A logic operation between f and g can be represented by OBDD manipulations as:

$$ite(x, F_1, F_0) \lozenge ite(y, G_1, G_0) = \begin{cases} ite(x, F_1 \lozenge G_1, F_0 \lozenge G_0) & \text{order}(x) = \text{order}(y) \\ ite(x, F_1 \lozenge g, F_0 \lozenge g) & \text{order}(x) < \text{order}(y) \\ ite(y, f \lozenge G_1, f \lozenge G_0) & \text{order}(x) > \text{order}(y) \end{cases} \quad (3)$$

where ◊ represnents a logic operation such as AND, OR, and NOT. Figure 1 illustrates the construction and manipulation steps of a Boolean function. For more details on using the operations of OBDD, please refer [9].

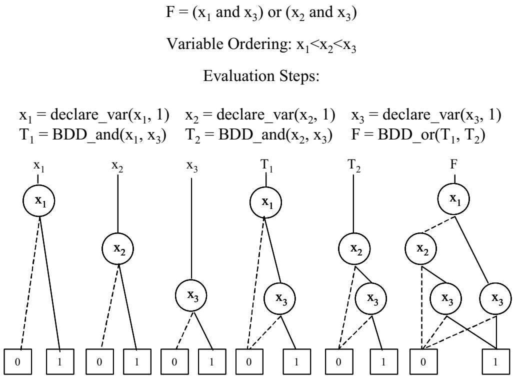
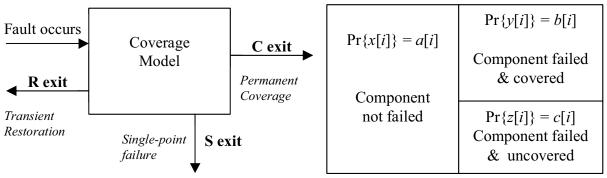


Figure 1. The OBDD generated from a Boolean equation.

2.3. Coverage Model

Figure 2(a) shows the general structure of a fault-coverage model representing a recovery process [12][13] initiated when a fault occurs. The entry point to the model signifies the occurrence of a fault, and the three exits (R, S, C) signify the 3 possible outcomes.

- If the offending fault is transient and can be handled without discarding any components, then the transient restoration exit (R) is taken.
- If the fault is determined to be permanent, and the offending component is discarded, then the permanent fault-coverage exit (C) is taken.
- If the fault by itself causes a system to fail, then the single-point failure exit (S) is taken.



(a) **(b)**

Figure 2. (a) General structure of a fault coverage model. (b) The event and probability space of component *i*.

The exit probabilities r_0, c_0, s_0 are required for the analysis of system reliability. The exits are a partitioning of the event space; thus the three exit probabilities sum to one, i.e. $(c_0 + s_0) = (1 - r_0)$. The r_0, c_0, s_0 can be determined by an appropriate fault coverage model [13]; for more details, see [6][8].

For the fault coverage model, each component is always in one of three states: $x[i]$, $y[i]$, $z[i]$. To determine the system reliability (unreliability), it is required to have $a[i]$, $b[i]$, $c[i]$ which represent the probabilities of component i associated respectively with the exits of the fault coverage model. Figure 2(b) shows the event space (and corresponding probability) representation of a component. Therefore,

$$
\begin{aligned}
a[i] &= \exp[-(1-r_{i0})\cdot\lambda_{i0}\cdot t] \\
b[i] &= \frac{c_{i0}}{c_{i0}+s_{i0}}\cdot\left[1-\exp[-(1-r_{i0})\cdot\lambda_{i0}\cdot t]\right] \\
c[i] &= \frac{s_{i0}}{c_{i0}+s_{i0}}\cdot\left[1-\exp[-(1-r_{i0})\cdot\lambda_{i0}\cdot t]\right] \qquad (4)
\end{aligned}
$$

where (r_{i0}, c_{i0}, s_{i0}) are the probabilities of taking (transient restoration, permanent coverage, single-point failure) exit in the coverage model, and λ_{i0} is the rate of occurrence of fault of component i. It should be noted that the effective failure rate λ_i and the effective coverage factor c_i of component i are

$$
\begin{aligned}
\lambda_i &\equiv (c_{i0}+s_{i0})\cdot\lambda_{i0} = (1-r_{i0})\cdot\lambda_{i0} \\
c_i &\equiv c_{i0}/(c_{i0}+s_{i0}) \qquad (5)
\end{aligned}
$$

Amari et al. [12] proposed an efficient algorithm, the SEA, to calculate the reliability of a system under the imperfect coverage model. The basic idea is shown in the following equation and could be easily proved [12] by using conditional probabilities.

System Unreliability (U_s) =
Pr{at least one uncovered failure}×
Pr{system failure | a uncovered failure}
+ Pr{no uncovered failure}×
Pr{system failure | no uncovered failure} (6)

Let Pr{no uncovered failure} = $\prod_{i\in S}(a[i]+b[i]) = P_u$, then Pr{at least one uncovered failure} = $1-P_u$. Also let Pr{system failure | no uncovered failure} = U_{cs}. Since Pr{system failure | at least one uncovered failure} is always equal to 1, we have

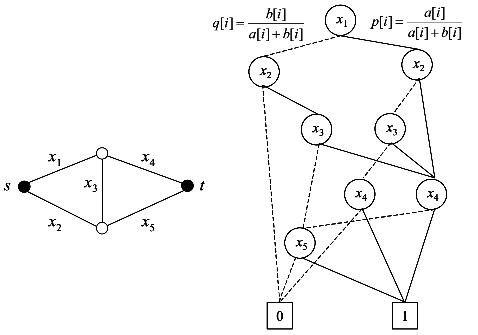
$$
\begin{aligned}
U_s &= (1-P_u)+P_u\cdot U_{cs} = 1-P_u\cdot R_{cs} \\
R_s &= 1-U_s = P_u\cdot R_{cs} \qquad (7)
\end{aligned}
$$

where R_s is the system reliability and R_{cs} is Pr{system success | no uncovered failure}.

Example 1:

For a terminal-pair network system, Kuo [14] proposed an efficient approach to determine the terminal-pair (from source node s to target node t) reliability based on edge expansion diagrams using OBDD. The main idea, which makes his approach very efficient, is that the OBDD of a given network is automatically constructed with mergence of isomorphic sub-problems during tracing all paths of the terminal-pair. Therefore, the system reliability is efficiently derived from OBDD.

Considering a bridge network as shown in Figure 3(a), Figure 3(b) shows the OBDD of this network system. Therefore, we get the conditional reliability, R_{cs}, of the network system by substituting the conditional reliability/ unreliability ($p[i]/q[i]$) for the reliability/unreliability of component i. Then, we can easily obtain the reliability of a network system subject to imperfect coverage from Equation (7). By this efficient integration, we don't need to solve the whole state's problem using Markov chains even when the network system is quite large and complex. In addition, using conditional probabilities, the computational complexity of this method is the same as that of the method for solving perfect coverage problems. □



(a) (b)

Figure 3. (a) A bridge network. (b) The OBDD of (a).

3. Multi-state Coverage Model

3.1. Multi-state Systems with Imperfect Coverage

Assume there are n modules in a system and module i has m_i states (i = 1,…,n). Depending upon the performance /capacity, we can arrange the states such that state m_i is a perfect state and state 1 is a failed state (the performance level decreases from state m_i to state 1). The ordering is not a constraint to apply the proposed algorithm, but it helps to use the existing algorithms for multi-state coherent system subject to perfect coverage model (PCM) as a part of an algorithm to solve the MSS problem subject to imperfect coverage model (IPCM). If imperfect coverage is introduced in the model, then each module will have an extra state; i.e. the total number of states in module i become m_i+1. Here, state 0 is the state corresponding to the uncovered failure of module i. Figure 4 shows the event and probability space of multi-state module i.

Assumption

A system subject to imperfect coverage will function satisfactorily as long as there exist system conditions that satisfy the system success requirements under the condition of perfect coverage and when there are no uncovered failures.

State Representation

state m_i perfect state of module i (highest performance level of module i)
state j state of a module at performance level j
state 1 state of a module at zero performance level (module failed and covered state)
state 0 module failed and uncovered state

Notation

x_i, x_i^I indicator variable of state of module i for [PCM, IPCM]; $x_i = l$ means modules i of PCM is in state l.

$x_{i:j}, x_{i:j}^I$ represents that module i of [PCM, IPCM] is at performance level j or above; i.e., $x_{i:j}, x_{i:j}^I$ are equivalent to $x_i \geq j, x_i^I \geq j$

$P_i(t,j)$ Pr{module i of PCM is in state j at time t}

$P_i^I(t,j)$ Pr{module i of IPCM is in state j at time t}

$P_i^c(t,j)$ Pr{module i of IPCM is in state j at time t | no uncovered failure in module i}

$R_i(t,j)$ Pr{module i of PCM is in state $\geq j$ at time t} $= \sum_{k=j}^{m} P_i(t,k) = \Pr\{x_{i:j}\}$

$R_i^I(t,j)$ Pr{module i of IPCM is in state $\geq j$ at time t} $= \sum_{k=j}^{m} P_i^I(t,k) = \Pr\{x_{i:j}^I\}$

$R_i^c(t,j)$ Pr{module i of IPCM is in state $\geq j$ at time t | no uncovered failure in module i}

$P_s(t,j)$ Pr{system of PCM is in state j at time t}

$P_s^I(t,j)$ Pr{system of IPCM is in state j at time t}

$P_s^c(t,j)$ Pr{system of IPCM is in state j at time t | no uncovered failure in module i}

$R_s(t,j)$ Pr{system of PCM is in state $\geq j$ at time t} $= \sum_{k=j}^{m} P_s(t,k)$

$R_s^I(t,j)$ Pr{system of IPCM is in state $\geq j$ at time t} $= \sum_{k=j}^{m} P_s^I(t,k)$

$R_s^c(t,j)$ Pr{system of IPCM is in state $\geq j$ at time t | no uncovered failure in module i}

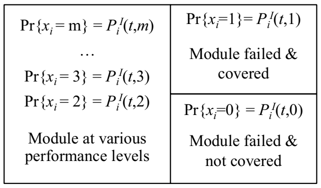

Pr{x_i = m} = $P_i^I(t,m)$
…
Pr{x_i = 3} = $P_i^I(t,3)$
Pr{x_i = 2} = $P_i^I(t,2)$
Module at various performance levels
Pr{x_i=1}= $P_i^I(t,1)$
Module failed & covered
Pr{x_i=0} = $P_i^I(t,0)$
Module failed & not covered

Figure 4. The probability space of multi-state module *i*.

Example 2 :

Figure 5 shows the combinatorial performance requirements of a system for being operational at performance level s. It includes three sub-requirement trees T_1, T_2, T_3. The event, $x_{i:j}$, of the tree means the minimum performance requirement for the system to operate at performance level s. That is, $x_{i:j}$ means module i needs to be operational at performance level j or above. For example, in Figure 5, the system will be at performance level s if every module i (i = 1, 2, 3) is operational at level 2 or above, or if any module i (i = 1, 2, 3) is operational at level 3 or above, or if module 1 is operational at level 4 or above, or both module 2 and module 3 are operational at level 4 or above.

From the definition of MSS [2], the probability of event $x_{i:j}$ is

$$\Pr\{x_{i:j}\} = \sum_{k=j}^{m} P_i(t,k) = R_i(t,j) \quad (8)$$

However, since "module 3 is operational at level 4 or above" implies "module 3 must be operational at level 3 or above", there exists dependency between Pr{$x_{3:3}$} and Pr{$x_{3:4}$}. We need to deal with the dependency problem in the probability calculation of combinatorial performance requirements.

□

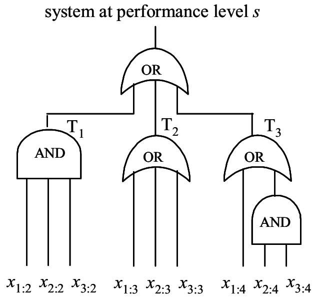

system at performance level s
OR
AND
OR
OR
AND
T_1
T_2
T_3
$x_{1:2}$ $x_{2:2}$ $x_{3:2}$ $x_{1:3}$ $x_{2:3}$ $x_{3:3}$ $x_{1:4}$ $x_{2:4}$ $x_{3:4}$

Figure 5. The combinatorial performance requirements of a multi-state system being operational at performance level *s*.

3.2. Reliability/Availability Evaluation of a Multi-state System

In this section, an algorithm similar to SEA for evaluating the system reliability of multi-state imperfect coverage is proposed. Using this method, the MSS with IPCM can be solved using the corresponding MSS subject to PCM.

System Reliability (R_s^I) =
Pr{no uncovered failure in system} ×
Pr{system success | no uncovered failure in system}
Pr{no uncovered failure in system} (P_u) =
$\prod_{i \in S}$ Pr{no uncovered failure of module i}
Pr{no uncovered failure of module i} =
$R_i^I(t, 1) = 1 - P_i^I(t, 0)$ (9)

Further,

$$R_i^c(t,j) = \Pr\{x_i^I \geq j \mid x_i^I > 0\} = R_i^I(t,j) / R_i^I(t,1) \quad (10)$$

This probability represents the conditional probability that module i of IPCM is in state j (i.e. performance level j). Therefore, Pr{system success | no uncovered failures in system} can be obtained by substituting $R_i(t, j)$ with $R_i^c(t, j)$ in the corresponding PCM model. Hence, the system reliability subject to imperfect coverage will be

$$R_s^I = P_u \times \left\{ \begin{array}{l} \text{Reliability of PCM with} \\ R_i(t,j) = R_i^c(t,j), P_i(t,j) = P_i^c(t,j) \end{array} \right\} \quad (11)$$

where $P_u = \prod_{i \in S} R_i^I(t,1)$, $R_i^c(t, j) = R_i^I(t, j)/R_i^I(t, 1)$, $P_i^c(t, j) = P_i^I(t, j)/R_i^I(t, 1)$.

It should be noted that the same algorithm is applicable for availability evaluation, but in this case the input-set to the algorithm should be derived using components availability models. Moreover, the state probabilities of MSS subject to IPCM can be found as follows:

$P_s^I(t,j) = P_u \times P_s^c(t,j)$

$P_s^c(t,j) = P_s(t,j)$ of PCM with $P_i(t,j) = P_i^c(t,j)$ (12)

Therefore, the step wise procedure of the proposed algorithm is as follows:

1. Read the state probabilities of all modules, i.e. $P_i^I(t, j)$ for $i = 1, \ldots, n; j = 0, 1, \ldots, m_i$.
2. For all i, find $R_i^I(t, 1) \equiv 1 - P_i^I(t, 0)$

3. Find $P_u = \prod_{i=1}^{n} R_i^I(t,1)$
4. Find the conditional probabilities of each module at every level;
   - $R_i^c(t, j) = R_i^I(t, j)/R_i^I(t, 1)$
   - $P_i^c(t, j) = P_i^I(t, j)/R_i^I(t, 1)$
5. Use these conditional probabilities to find the system reliability/availability (or the probability of a system state) at the required performance level of the corresponding MSS subject to PCM. Solve modular structures using modularization method. Use the proposed OBDD method to solve generic problems. Let this conditional probability be P.
   - Reliability $R_s^c(t, j)$: Find the reliability of MSS with PCM by substituting either $P_i^c(t, j)$ for $P_i(t, j)$ or $R_i^c(t, j)$ for $R_i(t, j)$
   - Availability $R_s^c(t, j)$: Find the availability of MSS with PCM by substituting either $P_i^c(t, j)$ for $P_i(t, j)$ or $R_i^c(t, j)$ for $R_i(t, j)$ (here $R_i^c(t, j)$ and $R_i(t, j)$ are corresponding availabilities).
   - System state probability $P_s^c(t, j)$: Find the probability of the state of MSS with respect to PCM by substituting either $P_i^c(t, j)$ for $P_i(t, j)$ or $R_i^c(t, j)$ for $R_i(t, j)$.
6. Find the system reliability/availability (or the probability of a system state) of MSS with IPCM using:
   Reliability/Availability = $P_u\, P$

Consider that the problem can be solved using general multi-state algorithms. The computational time is proportional to $O(m+1)^n$. Hence, the computational complexity is $O(m^n)$. However, we can't use modularization methods in this case. In our method, the advantage of using conditional probabilities makes it possible to apply this method for modular structures. Then the problem can be solved in linear time (in most cases). Without using conditional probabilities, we can't apply the solutions of modular structures.

The percentage reduction (reduction factor) by using our method is approximately $1-(\frac{m}{m+1})^n$. That means the reduction increases with n and decreases with m. But the difference $(m+1)^n$–m^n actually increases with m. However, this is a worst case situation without modular structures in a system. In general, our method is much more faster than existing methods.

Figure 6 shows the OBDDs of three sub-requirement trees in Example 2. As mentioned earlier, there exist some dependencies between the sub-requirement trees, and it generates dependency problems for the probability calculation in step 5 of our algorithm. Next section will show the methods to deal with the problems.

4. Multi-state OBDD

4.1. Multi-state Dependency Operation (MDO)

In a multi-state system with multi-state modules, different performance levels for the same module are represented by different nodes of the OBDD as shown in Figure 6. The operations on OBDD should be modified to deal with the dependency problem in probability calculation. We use of the methods in [15][16] to solve this problem.

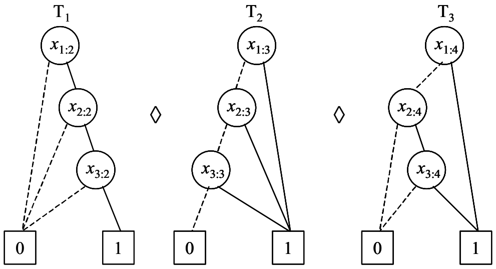


Figure 6. The OBDD of sub requirement tree T_1, T_2, T_3.

For example, $x_{i:k}$ means module i is operational at performance level k or above and $x_{i:l}$ means the same module i is operational at performance level l or above. If level k is greater than level l, then $x_{i:k} = 1$ implies $x_{i:l} = 1$, i.e. $x_{i:k} \subseteq x_{i:l}$. Therefore, there exist dependencies between $x_{i:k}$ and $x_{i:l}$. We need to deal with dependency in constructing the OBDD tree of a multi-state system. It means some branches should be cut during the multi-state dependency operation (MDO).

Lemma 1:

If performance level k is greater than performance level l, i.e. $x_{i:k} \subseteq x_{i:l}$, and the order of $x_{i:k}$ is smaller than $x_{i:l}$, then

$$\begin{aligned} & ite(x_{i:k}, K_1, K_0) \lozenge ite(x_{i:l}, L_1, L_0) \\ &\quad = E_k \lozenge E_l \\ &\quad = ite[x_{i:k}, K_1 \lozenge L_1, K_0 \lozenge E_l] \end{aligned} \qquad (13)$$

Proof: Since the two nodes belong to the same module, module i being operational at performance level k or above means module i must be operational at performance level l or above, i.e. $x_{i:k} = 1$ implies $x_{i:l} = 1$. Therefore,

$$\begin{aligned} & ite(x_{i:k}, K_1, K_0) \lozenge ite(x_{i:l}, L_1, L_0) \\ &\quad = E_k \lozenge E_l \\ &\quad = ite[x_{i:k}, (E_k \lozenge E_l)_{x_{i:k}=1}, (E_k \lozenge E_l)_{x_{i:k}=0}] \\ &\quad = ite[x_{i:k}, (E_k)_{x_{i:k}=1} \lozenge (E_l)_{x_{i:l}=1}, (E_k)_{x_{i:k}=0} \lozenge (E_l)_{x_{i:k}=0}] \\ &\quad = ite[x_{i:k}, K_1 \lozenge L_1, K_0 \lozenge E_l] \end{aligned}$$

The derivation use the relation $(E_l)_{x_{i:k}=0} = E_l$ since $x_{i:k} = 0$ is not relevant to E_l.

□

For example, in Figure 6, if the ordering of the nodes is $x_{1:4} < x_{1:3} < x_{1:2} < x_{2:4} < x_{2:3} < x_{2:2} < x_{3:4} < x_{3:3} < x_{3:2}$, after applying the MDO on the system, we get the result as shown in Figure 7. It should be noted that $x_{3:4}$ is automatically eliminated during MDO. This is because, from the sub-requirement tree T_2 in Figure 5, when module 3 is operational in performance level 3 or above, it makes the system meet the required performance level s. Hence, we don't need to consider if module 3 is operational in performance level 4 or above in sub-requirement tree T_3, i.e. the node $x_{3:4}$ disappears.

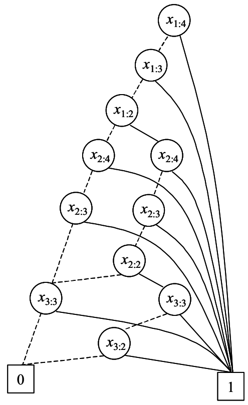


Figure 7. The OBDD of the system in Figure 5.

4.2. Dependency Probability Calculation

The traditional recursive algorithm can efficiently calculate the probability for an OBDD. However, for the OBDD tree of a multi-state system as shown in Figure 7, there exists dependency between a node and its child node if they belong to the same module but with different performance levels. Therefore, the traditional recursive algorithm should be modified to deal with the dependency problem in probability calculation.

Let k and l be two performance levels ($k > l$). Table 1 shows the rules of level algebra. The validness of these relationships can be easily verified. Note that $x_{i:k} = w_1 w_2 \dots w_k$, where w_h is the Boolean value (True/False) that represents module i to be (operational/failed) at performance h ($1 \le h \le k$).

Table 1. The rules of level algebra (*k* > *l*).

operation	value
1. $x_{i:k} \cdot x_{i:l}$	$x_{i:k}$
2. $x_{i:k} \cdot \bar{x}_{i:l}$	0
3. $\bar{x}_{i:k} \cdot \bar{x}_{i:l}$	$\bar{x}_{i:l}$
4. $\bar{x}_{i:k} \cdot x_{i:l}$	$x_{i:l} - x_{i:k}$

Therefore, for the 1st rule,

$$x_{i:k} \cdot x_{i:l} = (w_1 w_2 ... w_k) \cdot (w_1 w_2 ... w_l) = w_1 w_2 ... w_k = x_{i:k}$$

The physical meaning of this equation is that the requirement "module i is operational both at performance level k and l" is equivalent to the requirement "module i is operational at performance level k". For the 2nd rule,

$$x_{i:k} \cdot \bar{x}_{i:l} = (w_1 w_2 ... w_k) \cdot (\bar{w}_1 + \bar{w}_2 + ... + \bar{w}_l) = 0$$

The physical meaning of this equation is that the requirement "module i is operational at performance level k, but is not operational at performance level l" does not exist. For the 3rd rule,

$$\bar{x}_{i:k} \cdot \bar{x}_{i:l} = (1 - w_1 w_2 ... w_k) \cdot (\overline{w_1 w_2 ... w_l}) = \bar{x}_{i:l}$$

The physical meaning of this equation is that the requirement "module i is not operational both at performance level k and l" is equivalent to the requirement "module i is not operational at performance level l".

For the 4th rule, $\bar{x}_{i:k} \cdot x_{i:l}$ means that module i is operational at performance level l, but failes when the performance achieves level k, and we have

$$\bar{x}_{i:k} x_{i:l} = (1 - w_1 w_2 ... w_k) \cdot (w_1 w_2 ... w_l) = x_{i:l} - x_{i:k}$$

These rules in Table 1 are only applicable to variables or nodes belonging to the same module. The ordinary Boolean relationships hold for the indicator variables belonging to different modules since module i and j are s-independent.

Looking into the OBDD tree constructed from MDO as shown in Figure 7, we find that a 1-edge always connects two nodes that belong to different modules. However, for the 0-edge, there are two cases that must be treated differently:

1. The 0-edge linking nodes belong to different module.
2. The 0-edge linking nodes belong to the same module.

In case 1, we calculate the probability of the node using ordinary OBDD method since the nodes are s-independent to each other. However, in case 2, we should make some modifications in probability calculation.

Lemma 2:

For the nodes ($x_{i:k}$, $x_{j:l}$) belonging to different modules, if $G = ite\ (x_{i:k}, G_1, G_0)$ and $G_0 = ite\ (x_{j:l}, H_1, H_0)$, then

$$\begin{aligned} \Pr\{G\} &= \Pr\{x_{i:k} G_1 + \bar{x}_{i:k} G_0\} \\ &= \Pr\{x_{i:k}\} \cdot [\Pr\{G_1\} - \Pr\{G_0\}] + \Pr\{G_0\} \qquad (14) \end{aligned}$$

Proof: This is an ordinary OBDD equation of calculation.

□

Lemma 3:

For the nodes ($x_{i:k}$, $x_{i:l}$) belonging to the same module but with different performance levels, if $G = ite\ (x_{i:k}, G_1, G_0)$ and $G_0 = ite\ (x_{i:l}, H_1, H_0)$, then

$$\Pr\{G\} = \Pr\{x_{i:k}\} \cdot \left[\Pr\{G_1\} - \Pr\{H_1\}\right] + \Pr\{G_0\} \qquad (15)$$

Proof: Since the 1-edge branch of $x_{i:k}$ always links to the node which belongs to a different module, $\Pr\{x_{i:k}\}$ is independent of $\Pr\{G_1\}$, i.e. $\Pr\{x_{i:k}G_1\} = \Pr\{x_{i:k}\}\Pr\{G_1\}$. Therefore, applying the rules of level algebra in Table 1, we get

$$\begin{aligned} \Pr\{G\} &= \Pr\{x_{i:k} G_1 + \bar{x}_{i:k} G_0\} \\ &= \Pr\{x_{i:k} G_1\} + \Pr\{\bar{x}_{i:k} x_{i:l} H_1 + \bar{x}_{i:k} \bar{x}_{i:l} H_0\} \\ &= \Pr\{x_{i:k}\} \Pr\{G_1\} + \Pr\{(x_{i:l} - x_{i:k}) H_1 + \bar{x}_{i:l} H_0\} \\ &= \Pr\{x_{i:k}\} \cdot [\Pr\{G_1\} - \Pr\{H_1\}] + \Pr\{G_0\} \end{aligned}$$

□

Let us consider the reliability calculation of a multi-state system with imperfect coverage using OBDD. As depicted in Section 3, we first find the conditional probability $\Pr\{x'_{i:k}\}$ of each multi-state module i. Second we use $\Pr\{x'_{i:k}\}$ instead of the probability or reliability of module i, $\Pr\{x'_{i:k}\}$, to calculate the multi-state system conditional reliability R_{cmss} from Equations (14) and (15). Therefore, we have the following lemma.

Lemma 4:

If G = ite ($x_{i:k}$, G_1, G_0), G_0 = ite (Z, H_1, H_0), and the order of node $x_{i:k}$ is smaller than that of node Z, the probability of G is

$$\Pr\{G\} = \begin{cases} \Pr\{x'_{i:k}\}\left[\Pr\{G_1\} - \Pr\{G_0\}\right] + \Pr\{G_0\} \\ \quad \text{(if } x_{i:k} \text{ and } Z \text{ belong to different modules)} \\ \Pr\{x'_{i:k}\}\left[\Pr\{G_1\} - \Pr\{H_1\}\right] + \Pr\{G_0\} \\ \quad \text{(if } x_{i:k} \text{ and } Z \text{ belong to the same module)} \end{cases} \tag{16}$$

where $\Pr\{x'_{i:k}\}$ is the conditional reliability of module i being operational at performance level k or above given that no uncovered failure occurred in that module (or module i).

Therefore, the probability of the OBDD's root node representing the multi-state system conditional reliability R_{cmss} is obtained from Equation (16). Hence, we get the multi-state system reliability R_{MSS} by

$$R_{MSS} = R_{cmss} \times P_u \tag{17}$$

5. Examples

Let us consider a bridge network shown in Figure 3(a) of Example 1. Assuming redundancy techniques are used such that each link has a fault-tolerance scheme. Therefore, we can treat a link as a module with various link capacities or with various performance levels (i.e. a multi-state network system). Also the fault-coverage condition should be considered. The path function of the system is

$$F = x_1x_3x_5 + x_1x_4 + x_2x_5 + x_2x_3x_4$$

Case I– the basic requirement for the system being in a acceptable performance level is:

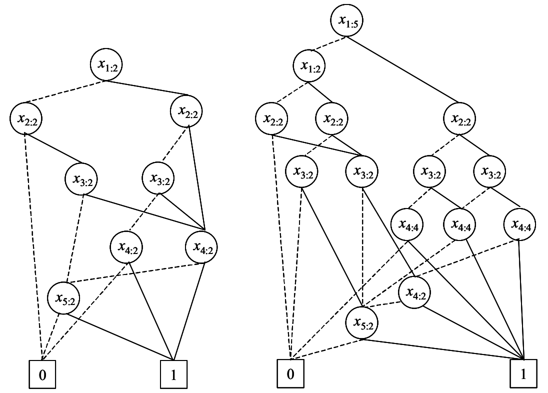
$$\Phi_{accept} = x_{1:2}x_{3:2}x_{5:2} + x_{1:2}x_{4:2} + x_{2:2}x_{5:2} + x_{2:2}x_{3:2}x_{4:2}$$

Case II– the path x_1x_4 is the backbone of the network and most of the dataflow run through the path, and the limitation of the requirement for the path is more strict. Therefore if x_1 needs to be at least in level 5 and x_4 needs to be at least in level 4 for path x_1x_4, the requirement for the system being in a good performance level is:

$$\Phi_{good} = x_{1:2}x_{3:2}x_{5:2} + x_{1:5}x_{4:4} + x_{2:2}x_{5:2} + x_{2:2}x_{3:2}x_{4:2}$$

Figure 8(a)(b) show the results of Φ_{accept} and Φ_{good} after applying MDO. Table 2 shows the parameters of a module obtained by using Markov techniques and assuming each module has 6 performance states including failed and uncovered state. If all modules in Figure 3(a) are identical, the system reliabilities of Φ_{accept} and Φ_{good} with different coverage factors are obtained from Equation (16) and (17) as shown in Figure 9. Figure 9 shows that Φ_{good} is less reliable than Φ_{accept}. That means we need to pay more on the system if we want to increase the reliability of Φ_{good} to be the same as that of Φ_{accept}. Figure 10 illustrates the system reliability decreases when the required level of path x_1x_4 increases. This means the higher the required performance level of a module, the more difficult the system satisfying a demand.

6. Conclusions



(a) The OBDD of Φ_{accept} (b) The OBDD of Φ_{good}

Figure 8. The results of Φ_{accept} and Φ_{good} after MDO.

Table 2: The parameters of individual module with exponential distribution given t = 20, failure rate λ_i = 0.015 and coverage factor c_i = 0.9.

j	$P^I(t,j)$	$P^c(t,j)$	$R^c(t,j)$
0	0.0259	——	——
1	0.2333	0.2395	1.0000
2	0.0823	0.0845	0.7605
3	0.2469	0.2535	0.6760
4	0.2469	0.2535	0.4225
5	0.1646	0.1690	0.1690

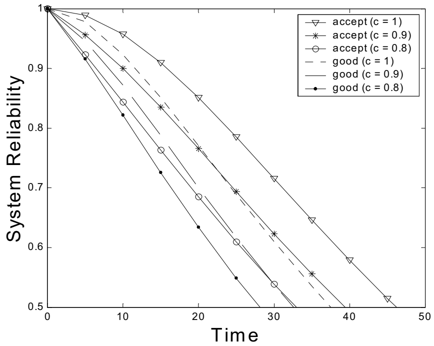


Figure 9. The system reliability of Φ_{accept} and Φ_{good} with different coverage factors c.

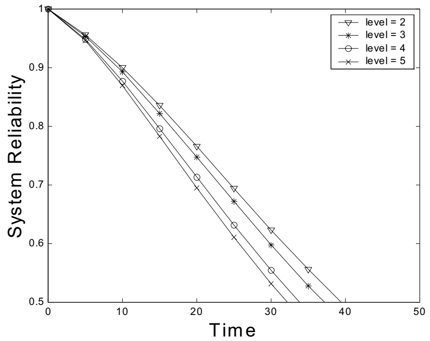


Figure 10. The system reliabilities of Φ_{good} with different required levels of path x_1x_4 and c = 0.9.

This paper has proposed an OBDD-based approach for the reliability evaluation of a multi-state system with combinatorial performance requirements subject to imperfect fault coverage. A new model for multi-state systems with imperfect fault coverage has also been proposed. It was shown that the algorithm used to evaluate the reliability can also be used to evaluate the availability of a system subject to imperfect fault coverage if the Markov process is applied to analyze the state transition behavior. Further, with the application of conditional probabilities, the time complexity of this method for reliability evaluation is the same as that without considering imperfect coverage.

In addition, an efficient integration of OBDD and modularization simplifies the problem further. The multi-state dependency operation (MDO) method handles the dependencies between the combinatorial performance requirements. Through the MDO method, some of the redundant nodes in OBDD are automatically eliminated. This means that we can simplify the combinatorial performance requirements using MDO. Moreover, our approach deals with the dependency problem of multi-state modules in probability calculation.

[2] and [17], with helpful comments from [2], have presented various performance measures related to multi-state systems. In order to compute these measures we need to find the reliability of a system at various performance levels. Therefore, the result of this paper can be integrated to find the performance measures of multi-state systems. This process is straightforward, and therefore, it is not discussed here explicitly [2].

This algorithm could be applied to complex systems such as fault-tolerant computer systems, variable link-capacities network systems, etc., since it generates the complete results more quickly and accurately even when there exist a number of dependencies such as shared loads (reconfiguration), degradation, common- cause failures and so on. Based on this approach, researches on sensitivity analysis, importance measures, failure frequency analysis or optimal design issues of multi-state systems will be the focus of our future works.

7. Acknowledgement

This research was supported by the National Science Council, Taiwan, R.O.C. under grant NSC 90-2213-E-002-113.

8. References

[1] J.D. Murchland, "Fundamental concepts and relations for reliability analysis of multistate systems", *Reliability and Fault Tree Analysis*, *Theoretical and Applied Aspects of System Reliability*, SIAM, 1975, pp. 581-618.

[2] J. Xue and K. Yang, "Dynamic reliability analysis of coherent multi-state systems", *IEEE Trans. on Reliability*, Vol.44, Dec. 1995, pp. 683-688.

[3] G. Levitin, "Incorporating Common-Cause Failure Into Nonrepairable Multistate Series-Parallel System Analysis", *IEEE Trans. on Reliability*, Vol. 50, No. 4, Dec. 2001, pp. 380-388.

[4] G. Levitin, A. Lisnianski, H. Beh-Haim, and D. Elmakis, "Redundancy optimization for series-parallel multi-state systems", *IEEE Trans. on Reliability*, Vol. 47, June 1998, pp. 165-172.

[5] J. Xue, "On multistate system analysis", *IEEE Trans. on Reliability*, Vol. R-34, Oct. 1985, pp. 329-337.

[6] J.B. Dugan, "Fault Tree and Imperfect Coverage", *IEEE Trans. on Reliability*, Vol. R-38, June 1989, pp. 177-185.

[7] S.V. Amari, J.B. Dugan, and R.B. Misra, "Optimal Reliability of Systems subject to Imperfect Fault-Coverage", *IEEE Trans. on Reliability*, Vol. 48, No. 3, Sep. 1999, pp. 275-284.

[8] S.A. Doyle, J.B. Dugan, F.A. Patterson-Hine, "A combinatorial Approach to Modeling Imperfect Coverage", *IEEE Trans. on Reliability*, Vol. 44, Mar. 1995, pp. 87-94.

[9] R.E. Bryant, "Graph-based algorithms for Boolean function manipulation", *IEEE Trans. on Computers*, Vol. C-35, Aug. 1986, pp. 677-691.

[10] A. Rauzy, "New algorithms for fault tree analysis", *Reliability Engineering and System Safety*, Vol. 40, 1993, pp. 203-211.

[11] R.M. Sinnamon and J.D. Andrews, "Improved efficiency in qualitative fault tree analysis", *Quality and Reliability Engineering Int'l.*, Vol. 13, 1997, pp. 293-298.

[12] S.V. Amari, J.B. Dugan, and R.B. Misra, "A separable method for incorporating imperfect fault-coverage into combinatorial models", *IEEE Trans. on Reliability*, Sep. 1999, pp. 267-274.

[13] J.B. Dugan and K.S. Trivedi, "Coverage modeling for dependability analysis of fault-coverage systems", *IEEE Trans. on Computers*, Vol. 38, June 1989, pp. 775-787.

[14] S.Y. Kuo, S.K. Lu, and F.M. Yeh, "Determining Terminal-Pair Reliability Based on Edge Expansion Diagrams Using OBDD", *IEEE Trans. on Reliability*, Vol. 48, Sep. 1999, pp. 234-246.

[15] X. Zang, H. Sun, and K.S. Trivedi, "Dependability Analysis of Distributed Computer Systems with Imperfect Coverage", *Proceedings of the Twenty-Ninth Annual International Symposium on Fault-Tolerant Computing*, 1999, pp. 330-337.

[16] X. Zang, H. Sun, and K.S. Trivedi, "A bdd-based algorithm for analysis of multi-state systems with multi-state components", Technical Report, 1998.

[17] S.V. Amari and R.B. Misra, "Comment on: Dynamic Reliability Analysis of Coherent Multistate Systems", *IEEE Trans. on Reliability*, Vol. 46, No. 4, Dec. 1997, pp. 460-461.