

# Dependability Analysis of Hierarchical Systems with Modular Imperfect Coverage

Liudong Xing; University of Virginia; Charlottesville, Virginia

Joanne Bechta Dugan; University of Virginia; Charlottesville, Virginia

Keywords: hierarchical systems (HS), modular imperfect coverage model (MIPCM), reliability, availability, fault tree (FT), multi-state Binary Decision Diagrams (MBDD)

## Abstract

We consider the dependability analysis of hierarchical fault tolerant computer based systems in this paper. The hierarchical nature of the system aids in fault coverage: if an undetected error escapes from one level of the system, it may be covered or tolerated at a higher level. A fault that remains uncovered through all levels of the hierarchy, and thus appears at the output, causes immediate system failure despite the remaining redundancy in the system. The difficulty in dependability analysis arises because the dependencies among components and/or modules across the different levels in the hierarchy introduced by such modular imperfect coverage have not been addressed in the solution. We offer a means by which to resolve this difficulty.

Our analysis approach uses a generalized coverage model for each component. This coverage model defines the conditional coverage probabilities associated with each level in the hierarchy. The resulting approach is applicable to Markov analyses and combinatorial methods such as FT, RBD, or MBDD for the analysis of both reliability and availability. We present the general approach in terms of a dynamic fault tree, and provide an analysis of a sample hierarchical system.

## Acronyms

CTMC	Continuous Time Markov Chain
DFT	Dynamic Fault Tree
HS	Hierarchical System
IPC	Imperfect Coverage
MBDD	Multi-State Binary Decision Diagram
MC	Markov Chain
MIPCM	Modular Imperfect Coverage Model
RBD	Reliability Block Diagram
SEA	Simple and Efficient Algorithm
w.r.t.	with respect to

## Introduction

Hierarchical systems (HS) are systems whose underlying architecture is characterized by multiple layers. Each layer of the hierarchy houses individual, separate modules and/or components. The hierarchical models can arise from either the system architecture or the model structure. In an HS, the upper-level's failure behavior often depends on the lower-level's failure behavior.

Figure 1 [adapted from ref. 2] is an example of a computer system whose system architecture exhibits a hierarchy with four levels or layers. The top or system layer is composed of three computing modules ( $CM_i$ ). Each computing module includes three memory modules ( $MM_{i,j}$ ), three identical CPU chips ( $CPUC_i$ ), and two identical port chips ( $PTC_i$ ). This layer is called the module layer. Each memory module is made up of ten identical memory chips ( $MC_{i,j}$ ) and one interface chip ( $IC_{i,j}$ ). This last layer is called the component layer.

The HS is decomposed by resolving the system layer into modules, and then resolving each successive module layer into components.

The following conditions must hold for the example system to be operational:

- at least eight of memory chips and one interface chip have to be unfailed for the memory module to be operational
- at least two memory modules, two CPU chips and one port chip must be operational to make each computing module operational
- at least two computing modules must be operational for the whole system to be operational

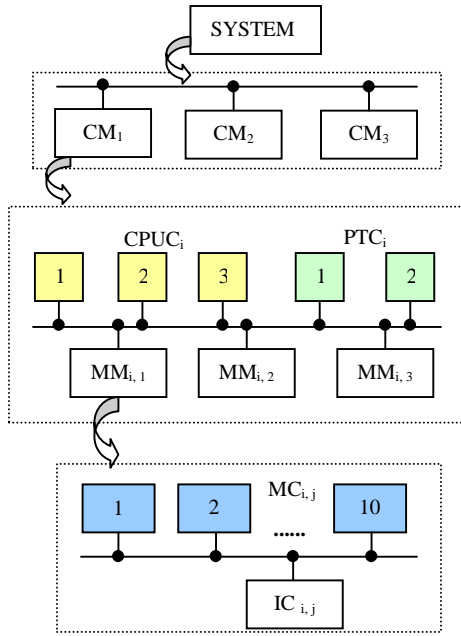


Figure 1 - A Sample Hierarchical System

Figure 2 is another example of an HS. This HS demonstrates how the hierarchical models can also arise from the analysis model structure. We use a reliability block diagram (RBD), a success oriented network describing the function of a system, in this example (ref. 4).

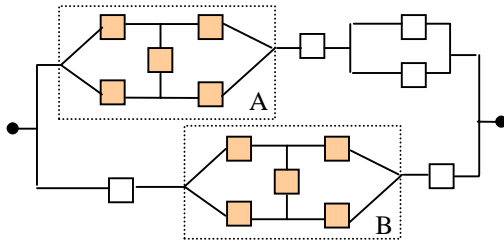


Figure 2 - System Model using RBD

Given that the two bridge modules A and B (composed of shaded boxes in Figure 2) are identical in structure, we can simplify the system model by two levels. The bridge subsystem specified by the lower level model (Figure 3) is replaced with a single node in the upper level model (Figure 4).

Dependability analysis is a key step in the design, analysis, and tuning of computer-based systems. Many safety-critical systems occurring in diverse applications such as avionics, nuclear power, railroads, and networks fall under the category of hierarchical systems. Such systems are typically designed with sufficient redundancy

to be fault tolerant. However, system failure can result if the system cannot adequately detect, locate, and recover from a fault despite the presence of fault tolerance mechanisms. Such an uncovered failure, also called a single-point failure, leads to the failure of the entire system even when there exists adequate redundancy. Therefore, the capability to model imperfect fault coverage precisely is crucial to the correct evaluation of a practical computer-based system.

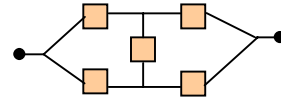


Figure 3 - Lower Level Model in RBD

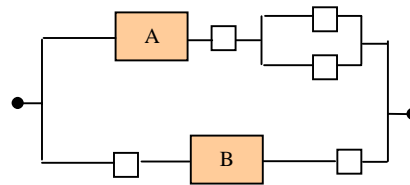


Figure 4 - Upper Level Model in RBD

However, existing analysis methods either didn't consider this important factor into their approaches to the solution or assumed conventional imperfect coverage (IPC). That is, the current approaches to IPC are *nothing* or *all* - either IPC is neglected or an uncovered failure kills the entire system. In an HS, however, an uncovered failure doesn't necessarily affect the whole system maliciously due to hierarchical recovery; the failure may just be propagated within a module in some level. More generally, though, an uncovered failure may propagate through multiple layers with different probabilities. We call such uncovered failure "modular imperfect coverage."

Modular imperfect coverage introduces dependencies among components and modules across different layers in hierarchical systems. These dependencies must be addressed in the analysis. The dependencies increase the complexity and difficulty of analysis.

We consider the problem of assessing the reliability and availability of an HS with modular imperfect coverage in this paper, and present a solution to the problem that has low computational complexity and which is easy to integrate into existing analytical methods.

## Modular Imperfect Coverage Model

The behavior of the system in the presence of faults is analyzed through a coverage model. In this section, we extend the traditional imperfect coverage model (refs. 5–7) to a modular imperfect coverage model (MIPCM) for HS. Let  $L$  denotes the number of layers of the hierarchical system. Figure 5 shows the general structure of a modular imperfect coverage model for a component in layer  $i$  in an HS. The entry point to the model signifies the occurrence of the fault. The  $(L-i+3)$  exits represent all possible outcomes. If the offending fault is transient and can be handled without discarding any component, then the transient restoration exit (labeled  $R$ ) is taken. The permanent coverage exit (labeled  $C$ ) denotes the determination of the permanent nature of the fault, and the successful isolation and removal of the faulty component. If the permanent coverage exit is reached, then a covered component failure is said to occur. When a single fault (by itself) brings down the layer  $i$  module to which the fault belongs, a single point failure (or uncovered failure) is said to occur. Further, if such an undetected fault is covered at layer  $i+1$ , the layer- $i$  single point failure exit (labeled  $S-i$ ) is reached; if the uncovered fault remains uncovered at layer  $i+1$ , but is tolerated at layer  $i+2$ , layer- $(i+1)$  single point failure exit (labeled  $S-(i+1)$ ) is reached; and if the fault remains uncovered through all levels of the hierarchy and thus brings down the module of the highest layer  $L$ , that is, the entire hierarchical system, then the layer- $L$  single-point failure exit (labeled  $S-L$ ) is reached. Uncovered component failures are said to occur for those  $(L-i+1)$  cases.

Several alternatives for the contents of the box used to represent the coverage model are possible (refs. 8, 9). However, it is required to refer only to the exit probabilities within the context of dependability analysis. The exits are mutually exclusive and complete. To be backward compatible with the traditional IPCM, we still define  $r$ ,  $c$ ,  $s$  to be the probability of taking the transient restoration, permanent coverage, single-point failure exits, respectively, given that a fault occurs (ref. 5).<sup>1</sup>

Further, we define  $p_k$  ( $k=i..L-1$ ) as the conditional probability that an undetected fault escapes from layer  $k$  but is covered at layer  $k+1$

<sup>1</sup>  $r+c+s=1$

conditioned on an uncovered failure occurring in layer  $i$ . Then  $s^*p_i$ ,  $s^*(1-p_i)^*p_{i+1}$ , ...,  $s^*\prod_{j=i}^{L-1}(1-p_j)$  will be the probability of taking layer  $i$ , layer  $i+1$ , ..., layer  $L$  single-point failure exit, respectively.

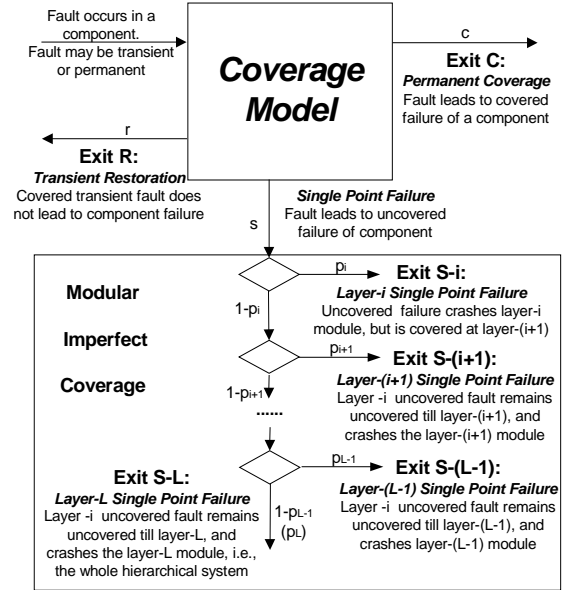


Figure 5 - General Structure of MIPCM for HS

Let  $q_{A_i}(t)$  be the failure function of component  $A$  that belongs to layer  $i$ ;  $n_{A_i}(t)$ ,  $c_{A_i}(t)$ ,  $u_{A_k}(t)$  denote the probability that the component  $A$  of layer  $i$  is operational, failed covered, and failed uncovered with respect to layer  $k$  ( $k=i..L$ ), respectively. Then, we have:

$$\begin{aligned} n_{A_i}(t) &= 1 - q_{A_i}(t) + q_{A_i}(t) * r_{A_i} \\ c_{A_i}(t) &= q_{A_i}(t) * c_{A_i} \\ u_{A_k}(t) &= \begin{cases} q_{A_i}(t) * s_{A_i} * p_{A_i}(k=i) & (1) \\ q_{A_i}(t) * s_{A_i} * \prod_{j=i}^{k-1} (1-p_{A_j}) * p_{A_k} (i < k < L) \\ q_{A_i}(t) * s_{A_i} * \prod_{j=i}^{L-1} (1-p_{A_j}) (k=L) \end{cases} \end{aligned}$$

## General Solution to MIPCM

Due to the layered peculiarity of HS, we need to use a *modular* and *hierarchical* decomposition method. We use dynamic fault trees (DFT) (ref.1) to decompose an HS. The solutions to each layer are combined hierarchically to obtain the whole system solution. Specifically, the

hierarchical system fault tree is composed of independent fault subtrees for each layer automatically. The layer subtrees are then solved hierarchically: a subtree is replaced by a single component in the parent layer subtrees whose probability of occurrence represents the occurrence probability of the layer subtree. Each layer subtree can further be decomposed using the modular DFT approach (ref.1), and then solved either as combinatorial model or as equivalent Markov chains.

Our general fault tree (FT) solution to MIPCM is shown in Figure 6. PFDEP denotes the “probabilistic functional dependency” gate. The PFDEP gate has a trigger input event (an uncovered failure of component), and one or more dependent events (the pseudo components for each possibility). When the trigger event associated with the component happens, the dependent (pseudo) inputs are then enforced to occur with certain probabilities. Each pseudo input will be distributed to corresponding layer subtree and make their contributions to that layer's failure.

The emphasis in this paper is on explaining our approach to the solution for a layer. We will explore and develop efficient approaches for analyzing the reliability and availability of each layer in an HS and addressing the dependencies introduced by MIPCM. We then use the modular DFT approach to combine the solutions to each layer hierarchically, thus obtaining the whole system solution .

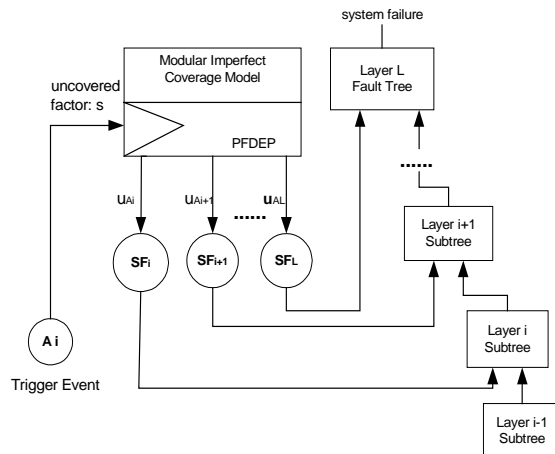


Figure 6 - FT Solution to MIPCM

### Reliability of Non-Repairable HS

The dependability measure of interest here for non-repairable HS is reliability. References 10

and 11 present a separable approach called SEA (simple and efficient algorithm) which incorporates conventional imperfect coverage into the solution for the reliability of a system. SEA separates the consideration of imperfect coverage from the combinatorics of the solution, thereby reducing the complexity of the solution. The problem with SEA is that it is only for conventional imperfect coverage in non-HS, i.e., a single-layer system, and not for modular imperfect coverage in HS. In this section, we will extend the SEA idea and propose another separable and efficient approach that incorporates modular imperfect coverage into a reliability solution for an HS.

**Separable Approach:** Consider a collection of mutually exclusive and complete events related to the failure of layer  $i$ :

- $E_1$  -- 1 or more components (including components belonging to layer  $i$  and those from the lower layers ( $1 \dots i-1$ )) experience layer- $i$  single-point failure
- $E_2$  -- no component experiences layer- $i$  single-point failure

Let  $N_i$  be the number of components in layer  $i$ . According to the “Total Probability Theorem,” for event  $E$ , the failure of layer  $i$ , we have:

$$UR_{layer i} = P[E] = \sum_{i=1}^2 P(E_i) * P(E/E_i) \quad (2)$$

where

$$P(E_i) = 1 - \prod_{A=1}^{N_i} [1 - u_{A_i}(t)] \prod_{k=1}^{i-1} \prod_{B=1}^{N_k} [1 - u_{B_k}(t)] \quad (3)$$

The first product item denotes the probability that no component in the layer  $i$  experiences layer- $i$  uncovered failure; the second product item denotes the probability that no component from all the lower layers experiences layer- $i$  uncovered failure. In addition,  $P(E_2) = 1 - P(E_1)$ ,  $P(E/E_1) = 1$ ,  $P(E/E_2) = Q_i$ .

As in SEA (refs. 10, 11),  $Q_i$  can be evaluated using any standard approach which ignores the concept of imperfect coverage such as reliability block diagram (RBD), binary decision diagram (BDD), and inclusion/exclusion.  $Q_i$  must be evaluated given that no component experiences layer- $i$  uncovered failure. Thus, the evaluation of  $Q_i$  should use a modified failure function  $\tilde{q}_{A_i}(t) = c_{A_i}(t) / [1 - u_{A_i}(t)]$ , instead of  $q_{A_i}(t)$  itself.

The unreliability of layer  $i$  in HS can then be calculated as:

$$UR_{layeri} = P[E] = P(E_1) + P(E_2)Q_i \quad (4)$$

The major contribution of this separable approach is that analysts can use their favorite software package that ignores modular imperfect coverage for reliability evaluation, and by slightly adjusting the input and output of the program, produce a result that incorporates the consideration of modular imperfect coverage. Also, the computational complexity is greatly reduced due to the separation of MIPCM from the combinatorics of the solution (ref. 13).

### Example

Consider the hierarchical computer system example shown in Figure 1 for the following parameter values (Table 1):

Comp.	Fail. Rate $\lambda$ ( $10^{-6} h^{-1}$ )			Coverage Factors $c$ and $p$
	CM1	CM2	CM3	
CPUC	0.6	1.2	1.8	$c=0.99, p_2=0.95$
PTC	0.6	1.2	1.8	$c=0.97, p_2=0.95$
IC	0.2	0.4	0.6	$c=1$
MC	0.2	0.2	0.2	$c=0.99, p_1=0.95, p_2=0.99$

Note that the probability of taking transient restoration exit  $r$  is ZERO for all components. The fault tree models shown in Figures 7 – 9 represent the failure criteria in each layer.

Assume mission time  $t = 200$  hours.

Solving the bottom layer fault tree model in Figure 9 using the separable approach presented in the last section, we can obtain the failure probability for each memory module (MM) in each of three computing modules (CM) as the following:  $P(MM_1)=4.3799e-05$ ,  $P(MM_2)=8.3796e-05$ ,  $P(MM_3)=1.2379e-04$ .

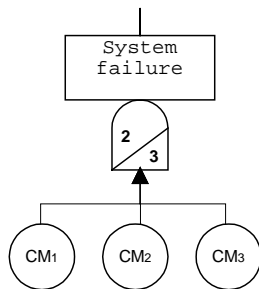


Figure 7 – Fault Tree Model for System Level

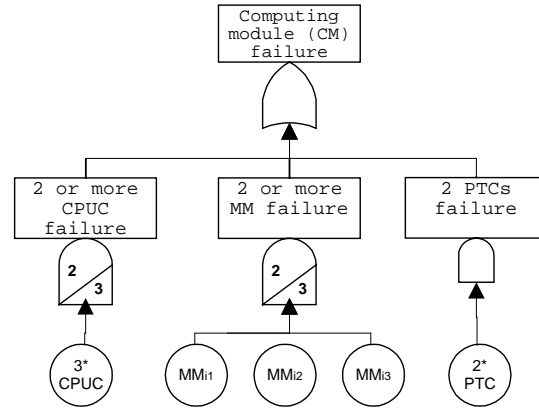


Figure 8 – Fault Tree Model for Middle Layer 2

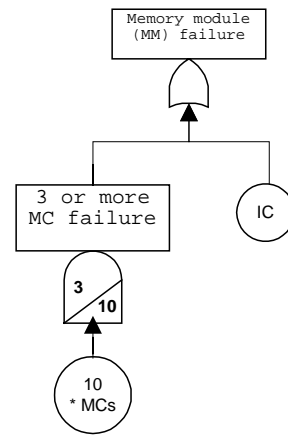


Figure 9 – Fault Tree Model for Bottom Layer 1

In the modular and hierarchical solution, each memory module is then replaced by a single component in the parent layer sub-fault tree as shown in Figure 8. Using the separable approach to solve the fault tree model in Figure 8, we obtain the following failure probabilities of each computing module:  $P(CM_1)=1.0519e-05$ ,  $P(CM_2)=2.096e-05$ ,  $P(CM_3)=3.1521e-05$ .

Lastly, we can obtain the unreliability of the whole system by solving the system level fault tree model in Figure 7:

$$UR_{sys} = 3.2468e-06$$

### Availability of Repairable HS

For hierarchical systems with independent repair, the dependability measure of interest becomes availability: the probability that the system is operating successfully at the instant time  $t$ . Most researchers represent the entire system as a

Markov model in order to analyze the repairable systems. However, Markov models have a disadvantage: their size grows exponentially as the size of the system increases. This rapid growth of states may lead to intractable models. In this section, we extend the multi-state concept (ref. 3), and propose two combinatorial approaches that can address both repair behavior and modular imperfect coverage in a very efficient way.

**Multi-State Concept:** Each component  $A$  in layer  $i$  in a HS can exist in one of the following states, and each state is represented by a Boolean variable (called multi-state variable)  $A_{is}$ ,  $s = O, C, U_i, U_{i+1}, \dots, U_L$ :

- Operational state:  $A_{iO}$ ;
- Covered failure state:  $A_{iC}$ ;
- Layer  $i$  uncovered failure state:  $A_{iU_i}$ ;
- Layer  $i+1$  uncovered failure state:  $A_{iU_{i+1}}$ ;
- .....
- Layer  $L$  uncovered failure state:  $A_{iU_L}$ .

The failure and repair behavior of each component in layer  $i$  can thus be modeled by a continuous-time Markov chain (CTMC) as shown in Figure 10.  $A_{is} = 1$  means that the component  $A$  in layer  $i$  is in the corresponding state  $s$ . Within the context of layer  $i$ , the three multi-state variables:  $A_{iO}, A_{iC}, A_{iU_i}$  can be related by  $\overline{A_{iO}} = A_{iC} + A_{iU_i}$ . This relation can also be represented in the BDD format as shown in Figure 11.

**Multi-State BDD:** BDD is the graphical representation of Boolean expressions. It has two sink nodes labeled with constants 0 (system operational) and 1 (system failed). The non-sink nodes are labeled with Boolean variables and have two outgoing edges. The left branch indicates the non-failure of the component represented by the nodes, and the right branch indicates the failure of the component. These edges are called “0-edge” and “1-edge,” respectively. The multi-state BDD (MBDD) is basically the same as the ordinary BDD depicted in the above; the only difference is that in the MBDD, non-sink nodes are labeled with multi-state variables. The 0-edge will indicate the component denoted by the nodes is not in the corresponding state (for instance, covered failure state if node is  $A_{iC}$ ), and the 1-edge will indicate the component is in the corresponding state. The variable ordering strategy is very important for

MBDD generation, because the size of MBDD heavily depends on the ordering (ref. 12).

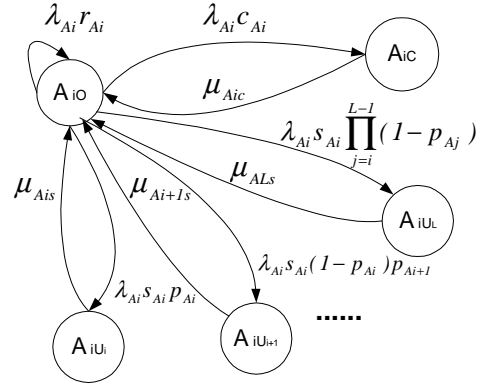


Figure 10 – CTMC for a Component in Layer  $i$

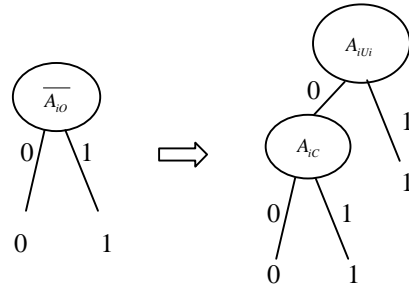


Figure 11 - Relation in BDD Format

The multi-state BDD for a layer in HS with MIPCM can be generated via the following steps:

- 1) Order the operational state variables using heuristics; construct ordinary BDD for layer  $i$  that doesn't consider imperfect coverage by applying the following logical operations on operational state variables, i.e.,  $\overline{A_{iO}}$ : let the if-then-else (*ite*) format for Boolean expression  $g$  and  $h$  be:

$$g = ite(x, g_{x=1}, g_{x=0}) = ite(x, G_1, G_2)$$

$$h = ite(y, g_{y=1}, g_{y=0}) = ite(y, H_1, H_2)$$

Let  $*$  denote any logic operation (AND/OR), then

$$g * h = \begin{cases} ite(x, G_1 * H_1, G_2 * H_2) - index(x) = index(y) & (5) \\ ite(x, G_1 * h, G_2 * h) - index(x) < index(y) \\ ite(y, g * H_1, g * H_2) - index(x) > index(y) \end{cases}$$

The same rules can be used for logic operation between sub-expressions until one of them becomes a constant 0 or 1.

- 2) Convert the ordinary BDD generated in step 1 to multi-state BDD that incorporate repair and MIPCM via the following two steps:
  - a) Replace all non-sink node  $\overline{A_{io}}$  in ordinary BDD by their corresponding nodes  $A_{iC}, A_{iUi}$  as shown in Figure 11
  - b) Because any layer  $i$  uncovered failure of components (including those belonging to layer  $i$  and those from the lower layers) will lead to the failure of layer  $i$ , use ordinary BDD OR operation to merge the variables that represent those uncovered failure, that is, all  $A_{iui}$  in layer  $i$  and all  $A_{kui}$  ( $k=1, \dots, i-1$ ) from lower layers.

**Evaluation of MBDD:** It's easy to see that the 1-edge always links variables of different components in the MBDD generated via the above method. For the 0-edge, however, there are two cases:

- linking two variables of different components
- linking two variables of the same component

For 1-edge or 0-edge linking variables of different components, the evaluation method is the same as in the ordinary BDD: consider a MBDD for  $G: G = ite(x, G_1, G_2) = xG_1 + \overline{x}G_2$ , then

$$P(G) = P(x)P(G_1) + [1 - P(x)]P(G_2) \quad (6)$$

where,  $P(G)$  is the unavailability w.r.t. the current sub-MBDD. And  $P(x)$  is the state occupation probability of  $x$  in the corresponding component Markov model.  $P(x)$  can be obtained via solving the CTMC shown in Figure 10.

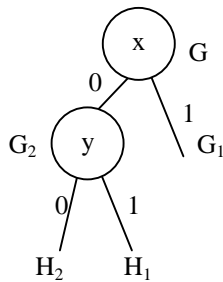


Figure 12 – A MBDD Branch in Evaluation

Next, consider the case where the 0-edge is linking two variables  $x$  and  $y$  that belong to the

same component. Assume  $x = A_{iUi}, y = A_{iC}$  for layer  $i$  in the MBDD. Because they represent the two failed states of the same component  $A$ ,  $x$  and  $y$  are not independent of each other; such dependence must be addressed in the solution.

Consider a MBDD branch for  $G$  shown in Figure 12, and

$$\begin{aligned} G &= ite(x, G_1, G_2) = xG_1 + \overline{x}G_2; \\ G_2 &= ite(y, H_1, H_2) = yH_1 + \overline{y}H_2 \end{aligned} \quad (7)$$

Because  $\overline{xy} = \overline{A_{iui}A_{iC}} = A_{iC} = y$  and  $\overline{y} = (\overline{x} + x)\overline{y} = \overline{x}y + x\overline{y} = \overline{x}y + A_{iui}\overline{A_{iC}} = \overline{x}y + A_{iui} = \overline{x}y + x$ , we have

$$\begin{aligned} P(G_2) &= P[yH_1 + \overline{y}H_2] = P[\overline{xy}H_1 + (\overline{x}y + x)H_2] \\ &= P[\overline{xy}H_1 + \overline{x}yH_2] + P[xH_2] \end{aligned}$$

$$\text{and then } P[\overline{xy}H_1 + \overline{x}yH_2] = P(G_2) - P(xH_2).$$

Hence,

$$\begin{aligned} P[G] &= P[xG_1 + \overline{x}G_2] = P[xG_1] + P[\overline{x}(yH_1 + \overline{y}H_2)] \\ &= P[xG_1] + P(G_2) - P(xH_2) \end{aligned}$$

As discussed before,  $G_1$  and  $H_1$  are independent of  $x$  because the 1-edge in MBDD always links variables of different components. Apparently,  $H_2$  cannot contain variables that represent the same component  $A$  as both  $x$  and  $y$  do; that is,  $H_2$  is also independent of  $x$ . Therefore, for 0-edge linking variables of the same component case, the evaluation will be

$$P[G] = P(x)[P(G_1) - P(H_2)] + P(G_2) \quad (8)$$

In summary, the recursive evaluation algorithm of the MBDD for layer  $i$  is:

- If  $x, y$  belong to different components  $P(G) = P(x)P(G_1) + [1 - P(x)]P(G_2)$
- If  $x, y$  belong to the same component  $P[G] = P(x)[P(G_1) - P(H_2)] + P(G_2)$
- Exit condition:
  - If  $G = 0, P(G) = 0$
  - If  $G = 1, P(G) = 1$
- If  $x$  is the root node, then  $P(G)$  will give the unavailability of the layer  $i$  in HS

**Availability Evaluation Based on MBDD:** So far, we can give the main procedure to evaluate the unavailability of one layer in HS with MIPCM:

1. For each component in the layer  $i$ , construct the CTMC as in Figure 10, and solve it to

obtain the state occupation probability for each state  $P(A_{i0}), P(A_{iC}), P(A_{iU_i}), P(A_{iU_{i+1}}), \dots, P(A_{iUL})$ . Note that in many applications, only the long-run steady state probabilities are of interest; that is, the values of  $P(A)$  as time  $t \rightarrow \infty$ . Thus, the system availability evaluated based on those steady state probabilities is steady state availability.

2. Construct the multi-state BDD for the layer  $i$  using the method presented in the *Multi-State BDD* section.
3. Traverse the resulted MBDD using the algorithm in the *Evaluation of MBDD* section to calculate the unavailability of layer  $i$

**Separable Availability Approach:** Note that in each layer MBDD, 1-edges of the nodes that represent layer- $i$  uncovered failure always connect to sink node 1 because the layer- $i$  uncovered failure leads to the immediate failure of layer  $i$ . Based on this fact, the separable approach presented in the *Reliability of Non-repairable HS* section can still be applied to separate the consideration of layer- $i$  uncovered failure from the combinatorics of the solution to evaluating the unavailability of a layer in the hierarchical system. Specifically, the layer  $i$  unavailability can be separated into two parts: layer- $i$  uncovered failure probability denoted as  $UA_{uncovered}$ , and covered failure part  $UA_{covered}$ . Based on ‘‘Total Probability Theorem,’’ the unavailability of layer  $i$  can be found using:

$$UA_{layer i} = UA_{uncovered} + (1 - UA_{uncovered}) \times UA_{covered} \quad (11)$$

where

- $UA_{uncovered} = 1 - \prod_{A=1}^{N_i} (1 - P(A_{iAi})) \times \prod_{k=1}^{i-1} \prod_{B=1}^{N_k} (1 - P(B_{kBi}))$
- $UA_{covered}$  should be evaluated given that no component experiences layer- $i$  uncovered failure. Hence, before calculating  $UA_{covered}$ , we should modify the covered failure state occupation probability  $P(A_{iC})$  to a conditional probability  $\tilde{P}(A_{iC})$  conditioned on there being no layer- $i$  uncovered failure occurring:

$$\tilde{P}(A_{iC}) = \frac{P(A_{iC})}{1 - P(A_{iU_i})} \quad (12)$$

$UA_{covered}$  can then be evaluated using any conventional combinatorial approaches that

ignore imperfect coverage, such as BDD and RBD. We provide the BDD evaluation in the following example:

- 1) Obtain  $\tilde{P}(A_{iC})$  for each component in layer  $i$  using equation 12.
- 2) Order variables  $A_{iC}$  using heuristics and construct ordinary BDD by applying ordinary logical operation (eq. 5) on the  $A_{iC}$ s.
- 3) Evaluate  $UA_{covered}$  recursively, using eq. 6, from the BDD using modified state occupation probability generated in 1.

Lastly, combine the solutions to each layer’s unavailability hierarchically using the DFT *modular* and *hierarchical* decomposition method to obtain the unavailability of the whole hierarchical system.

**Revisiting Earlier Example:** We will use the example given in the *Reliability of Non-repairable HS* section to explain these ideas. We will use the failure parameters and coverage factors listed in Table 1. The repair rates for each component upon reaching different kinds of failure states are shown in Table 2. Note that the rates are the same in the three computing modules, and that we use the module name as the index of state variables. For example,  $A_{UMM}$  denotes the uncovered failure state w.r.t. the memory module;  $A_{UCM}$  denotes the uncovered failure state w.r.t. the computing module; and  $A_{U_{sys}}$  denotes the uncovered failure state w.r.t. the whole system.  $A_C$  still represents the covered failure state. We analyze the example using the separable availability approach.

Table 2 – Repair Parameters For Example (X denotes that this parameter is not necessary)

Comp.	Repair Rate $\mu$ ( $10^{-4} \text{ h}^{-1}$ )			
	$A_C$	$A_{UMM}$	$A_{UCM}$	$A_{U_{sys}}$
CPUC	12	X	10	8
PTC	16	X	10	8
IC	30	8	6	4
MC	8	6	4	2

According to the algorithm presented in the preceding sections, we construct a CTMC for the memory and interface chips in the bottom layer, and a CTMC for the CPU and port chips and in the middle layer as shown in Figures 14 and 15, respectively. Based on the parameters provided in Table 1 and Table 2, we can obtain the steady state occupation probabilities for each

component by solving a set of balance equations for those CTMCs. The steady state occupation probabilities are different since the failure rates are different for each component in three computing modules (CM). Tables 3-5 give the steady state occupation probabilities for each component in the three CMs, respectively.

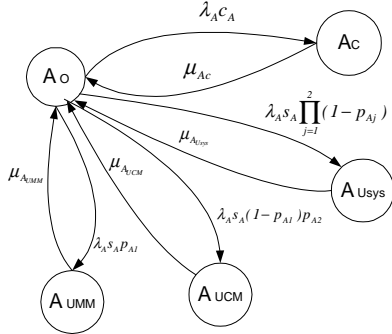


Figure 14 – CTMC for MC/IC in Bottom Layer

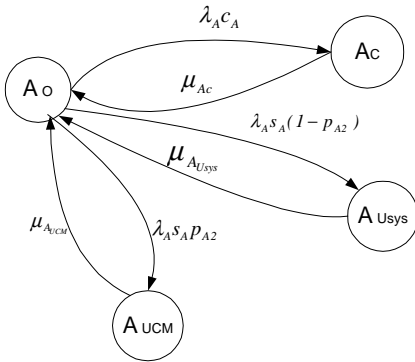


Figure 15 – CTMC for CPUC/PTC

Table 3 – State Occupation Prob. in CM1

Comp-onent	State occupation prob. in CM1			
	P(A <sub>c</sub> )	P(A <sub>UMM</sub> )	P(A <sub>UCM</sub> )	P(A <sub>Usys</sub> )
CPUC	4.9475e-04	X	5.6971e-06	3.7481e-07
PTC	3.6361e-04	X	1.7093e-05	1.1246e-06
IC	6.5996e-05	0	0	0
MC	2.4744e-04	3.1659e-06	2.4744e-07	4.9987e-09

Table 4 – State Occupation Prob. in CM2

Comp-onent	State occupation prob. in CM2			
	P(A <sub>c</sub> )	P(A <sub>UMM</sub> )	P(A <sub>UCM</sub> )	P(A <sub>Usys</sub> )
CPUC	9.8901e-04	X	1.1389e-05	7.4925e-07
PTC	7.2694e-04	X	3.4174e-05	2.2483e-06
IC	1.3198e-04	0	0	0

MC	2.4744e-04	3.1659e-06	2.4744e-07	4.9987e-09
----	------------	------------	------------	------------

Table 5 – State Occupation Prob. in CM3

Comp-onent	State occupation prob. in CM3			
	P(A <sub>c</sub> )	P(A <sub>UMM</sub> )	P(A <sub>UCM</sub> )	P(A <sub>Usys</sub> )
CPUC	0.0015	X	1.7074e-05	1.1233e-06
PTC	0.0011	X	5.1241e-05	3.3711e-06
IC	1.9796e-04	0	0	0
MC	2.4744e-04	3.1659e-06	2.4744e-07	4.9987e-09

We construct an ordinary BDD for each layer using the covered failure state variables  $A_c$ , and then evaluate the generated BDD using modified state occupation probabilities calculated from equation 12. The evaluation result is then combined with the layer uncovered failure probability using equation 11 to obtain the unavailability of this layer.

By solving the bottom layer FT using the separable approach, we obtain the steady state unavailability for the memory module (MM) in the middle layer for each CM.

MM	Steady State Unavailability		
	CM1	CM2	CM3
	9.7654e-05	1.6364e-04	2.2961e-04

By solving the middle layer FT containing MM whose occupation probabilities are the steady state unavailabilities obtained in last step, we have the steady state unavailability for each of the computing modules (CMs) in the top layer.

Steady State Unavailability		
CM1	CM2	CM3
5.4645e-05	1.0853e-04	1.6428e-04

By solving the top layer FT that involves the three CMs, we reach the final result – the steady state unavailability for the entire hierarchical system as:

$$UA_{sys} = 2.0413e-05$$

### Conclusions

This paper analyzes non-repairable HS reliability and repairable HS availability. As we have shown through examples, our separable methods have low computational complexity and are easy to implement. Our methods also provide the

means for incorporating MIPCM into both reliability and availability analysis.

## References

1. R. Gulati and J. B. Dugan. A Modular Approach for Analyzing Static and Dynamic Fault Trees. Proceedings of Annual Reliability and Maintainability Symposium, January 1997
2. V. Sune and J. A. Carrasco. A Failure-Distance Based Method to Bound the Reliability of Non-Reparable Fault-Tolerant Systems without the Knowledge of Minimal Cutsets. IEEE Transaction on Reliability.
3. X. Zang, H. Sun, and K. S. Trivedi. Dependability Analysis of Distributed Computer Systems with Imperfect Coverage. The 29th Annual International Symposium on Fault-Tolerant Computing, pages 330 - 337, 1999.
4. A. Hoyland and M. Rausand. System Reliability Theory. A Wiley-Interscience Publication, 1994.
5. S. A. Doyle, J. B. Dugan, and A. Patterson-Hine. A Combinatorial Approach to Modeling Imperfect Coverage. IEEE Transactions on Reliability, pages 87-94, March 1995.
6. J. B. Dugan. Fault Trees and Imperfect Coverage. IEEE Transactions on Reliability, 38(2):177 - 185, June 1989.
7. J. B. Dugan, S. A. Doyle, and L. L. Pullum. New Approaches to Fault Tree Analysis. Communication in Reliability, Maintainability, Supportability and Logistics, July 1996.
8. W. Bouricious, W. Carter, and P. Schneider. Reliability Modeling Techniques for Self-Repairing Computer Systems. Proc. 24th Ann. Nat'l Conf, pages 295-309, 1969.
9. J. B. Dugan, K. S. Trivedi, M. K. Smotherman, and R. M. Geist. The Hybrid Automated Reliability Predictor. AIAA Journal of Guidance, Control and Dynamics, 9(3):319-331, May- June 1986.
10. S. V. Amari, J. B. Dugan, and R. B. Misra. A Separable Method for Incorporating Imperfect Coverage in Combinatorial Model. IEEE Transaction on Reliability, 48(3), September 1999.
11. L. Xing and J. B. Dugan. Analysis of Generalized Phased-Mission Systems Reliability, Performance and Sensitivity. IEEE Transaction on Reliability, 2001 (Accepted).
12. R. E. Bryant. Graph-based Algorithms for Boolean Function Manipulation. IEEE Trans. on Computer, c-35(8):677-691, August 1986.
13. L. Xing. Dependability Modeling and Analysis of Hierarchical Computer-Based Systems. Ph.D. Dissertation, Electrical and Computer Engineering, University of Virginia, August 2001 (Expected).

## Biography

Liudong Xing, Ph.D. candidate, Department of Electrical & Computer Engineering, 351 McCormick Road, PO Box 400743, University of Virginia, Charlottesville, VA 22904-4743 telephone - (804) 924-1416 fax - (804) 924-8818 email - [lx6f@virginia.edu](mailto:lx6f@virginia.edu)

Liudong Xing received her B.S. degree in Computer Science from ZhengZhou University, China, in 1996. She was awarded the M.S. degree in Electrical Engineering from the University of Virginia in January, 2000. She is now a Ph.D. candidate in the Department of Electrical & Computer Engineering at the University of Virginia. Her major interests focus on dependability modeling and analysis of hierarchical computer-based systems, phased-mission systems and networks. She is a student member of IEEE.

Joanne Bechta Dugan, Ph.D., Professor, Department of Electrical & Computer Engineering, 351 McCormick Road, PO Box 400743, University of Virginia, Charlottesville, VA 22904-4743, USA, telephone - (804) 982-2078, fax - (804) 924-8818 email - [jbd@virginia.edu](mailto:jbd@virginia.edu)

Joanne Bechta Dugan is currently Professor of Electrical and Computer Engineering at the University of Virginia. She has performed and directed research on the development and application of techniques for the analysis of computer systems which are designed to tolerate hardware and software faults. Her research interests thus include hardware and software reliability engineering, fault tolerant computing, and mathematical modeling using dynamic fault trees, Markov models, Petri nets, and simulation. Dr. Dugan is an IEEE Fellow, was Associate Editor of the IEEE Transactions on Reliability for 10 years, and is currently Associate Editor of the IEEE Transactions on Software Engineering.

## Acknowledgements

This work was supported by NASA Langley Research Center under contract NAS1-99098. We thank Susan Donohue at University of Virginia for her comments and suggestions on the improvement of this paper