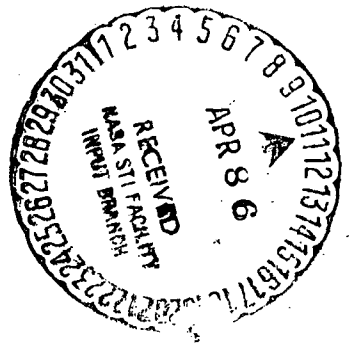
CSDL-T-897

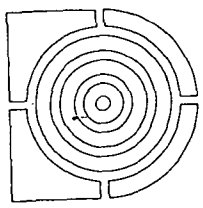
A HIERARCHICAL APPROACH TO RELIABILITY MODELING OF FAULT-TOLERANT SYSTEMS

by

William E. Gossman

January 1986





The Charles Stark Draper Laboratory, Inc.

Cambridge, Massachusetts 02139

A Hierarchical Approach to Reliability Modeling of Fault-Tolerant Systems

by

William Elting Gossman
B.S., Cornell University
(1984)

SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE
DEGREE OF

MASTER OF SCIENCE IN
AERONAUTICS AND ASTRONAUTICS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

January 1986

Signature of Author_____
Department of Aeronautics and Astronautics
January 14, 1986

Certified by_____
Dr. Richard S. Schabowsky, Jr.
Technical Supervisor, CSDL

Certified by_____
Professor Bruce K. Walker
Faculty Thesis Supervisor

Accepted by_____
Professor Harold Y. Wachman
Chairman, Departmental Graduate Committee

A HIERARCHICAL APPROACH TO RELIABILITY MODELING OF FAULT-TOLERANT SYSTEMS

by

William Elting Gossman

Submitted to the Department of Aeronautics and Astronautics on January 14, 1986, in partial fulfillment of the requirements for the Degree of Master of Science in Aeronautics and Astronautics

ABSTRACT

A methodology for performing fault-tolerant system reliability analysis is presented. The method decomposes a system into its subsystems, evaluates event rates derived from the subsystem's conditional state probability vector and incorporates those results into a hierarchical Markov model of the system. This is done in a manner that addresses failure sequence dependance associated with the system's redundancy management strategy. The method is derived for application to a specific system definition. Results are presented that compare the hierarchical model's unreliability prediction to that of a more complicated standard Markov model of the system. The results for the example given indicate that the hierarchical method predicts system unreliability to a desirable level of accuracy while achieving significant computational savings relative to a component-level Markov model of the system.

Thesis Supervisor: Dr. Bruce K. Walker
Title: Assistant Professor of Aeronautics and Astronautics

ACKNOWLEDGMENTS

Now that I know the light at the end of the tunnel is not a train comming at me, I want to acknowledge some exceptional people.

I am grateful to Dr. Richard Schabowsky for his thoughtful supervision, and for being one of the most pleasant people I have ever worked with.

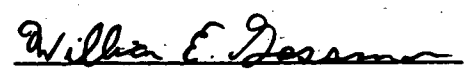
To Prof. Bruce Walker for his extraordinary insight and diligence. He should not go unrewarded now or in the future.

To the Harpers: Rick and Bud. The former for much assistance, both personally and technically. The latter for never biting a hand he knew was mine.

To Sam and Lisa, thanks for everything.

And finally to my parents. Did pretty good, didn't we?

This document was prepared at the Charles Stark Draper Laboratory with independent research and development funds under Project 19413.



William E. Gossman

TABLE OF CONTENTS

CHAPTER	Page
CHAPTER 1. Introduction	6
1.1 Background	6
1.2 Motivation	7
1.3 Methodology	10
1.4 Organization	12
CHAPTER 2. A Hierarchical Approach to Reliability Modeling	14
2.1 Outline of Hierarchical Model Development	14
2.2 Definitions	16
2.2.1 Definition of Terms	16
2.2.2 Definition of a Hierarchical Markov State	23
2.3 Sample Architecture	24
2.4 Hierarchical Model Development for Sample Architecture	27
2.4.1 Probability Density Function Determination	30
2.4.2 Conditional Probability Derivation	38
2.4.3 Initial Conditions	40
2.4.4 Calculation of Subsystem Event Rates	41

CHAPTER 3. Implementation and Simulation 44

3.1 Simulation Implementation 44

3.1.1 Solution of Event and Exact Models 44

3.1.2 Solution of Hierarchical Model 49

3.1.3 Program Outline 53

CHAPTER 4. Results 55

4.1 Computational Efficiency 55

4.2 Two Subsystem Results 61

4.2.1 Similar Subsystems 61

4.2.2 Dissimilar Subsystems 62

CHAPTER 5. Summary and Conclusions 73

5.1 Summary 73

5.2 Topics for Further Research 74

5.3 Conclusion 75

Appendix Page

Appendix A. Time-Varying Component Failure Rates 77

List of References 80

CHAPTER 1. INTRODUCTION

1.1 Background

Analytic reliability modeling is required to support the design and validation of highly reliable fault-tolerant systems (Ref{1}). The most accurate method for evaluating system reliability involves the life testing of many systems. Reliability statistics are subsequently derived from the test data. However, for large, highly reliable complex systems this is impractical due to the time and expense involved in testing such a system. Consequently it is necessary to analytically model the reliability of large highly reliable systems.

There are two general approaches to system reliability modeling, both of which rely upon knowledge of component failure rates (that <u>can</u> be found through life testing). First, there are combinatorial methods (Ref{2},{3}) in which component reliabilities and unreliabilities are used in conjunction with an enumeration of failure events to arrive at a system reliability prediction. Secondly, there is the Markov model approach (Ref{4}). This approach associates failure combinations with states of a Markov chain wherein component failure rates define transition rates.

1.2 Motivation

Both approaches encounter difficulties with the evaluation of large complex systems. A common combinatorial method is the event space method. This method utilizes an enumeration of combinations of failed and unfailed components. Each of these combinations defines the condition of every component in the system so that combinations are mutually exclusive and hence the probabilities are additive. For a large complex system, the large number of components will lead to many complex terms, and thus a large algebraic expression to be evaluated.

This situation can be mitigated somewhat through the use of structural decomposition (Ref{5}). Structural decomposition consists of dividing a system into smaller independent subsystems (redundancy management is local to the subsystem), analyzing the subsystems and then combining the subsystem results combinatorially to obtain the results for the system. However, this approach breaks down when the time sequence of events is an issue.

There are two types of sequence dependencies that arise in the evaluation of fault-tolerant systems. First, there are time-ordered event sequences associated with false alarms. This problem has been investigated by Luppold et al (Ref{6}). Second there are time-ordered event sequences that result from the system's redundancy management policy.

In particular, Schabowsky et al (Ref{5}) show how system reconfiguration can introduce time-ordered event sequences which greatly complicate a combinatorial approach.

In order to demonstrate the problem of time ordering of events resulting from a reconfiguration strategy, the unreliability of an example system (described in Chapter 2) is examined using three methods. These methods are: the event-space method, a component-level Markov model, and the hierarchical approach that is explored in this study. The results in Figure 1.1 clearly show that the absence of time-ordering considerations in the combinatorial analysis leads to a conservative unreliability prediction. This is due to the fact that some operational states are actually considered non-operational states in the event space method while they are accurately represented by the other two methods.

Although a Markov model approach conveniently handles time ordered events, this approach encounters difficulty when a system with a large number of components is considered. With many components (and thus many combinations of failures to be considered) the Markov model may have a very large state space. A large state space translates into a large system of differential equations that must be solved to produce an unreliability prediction. The proposed hierarchical approach mitigates the problem of state proliferation while still addressing the problem of

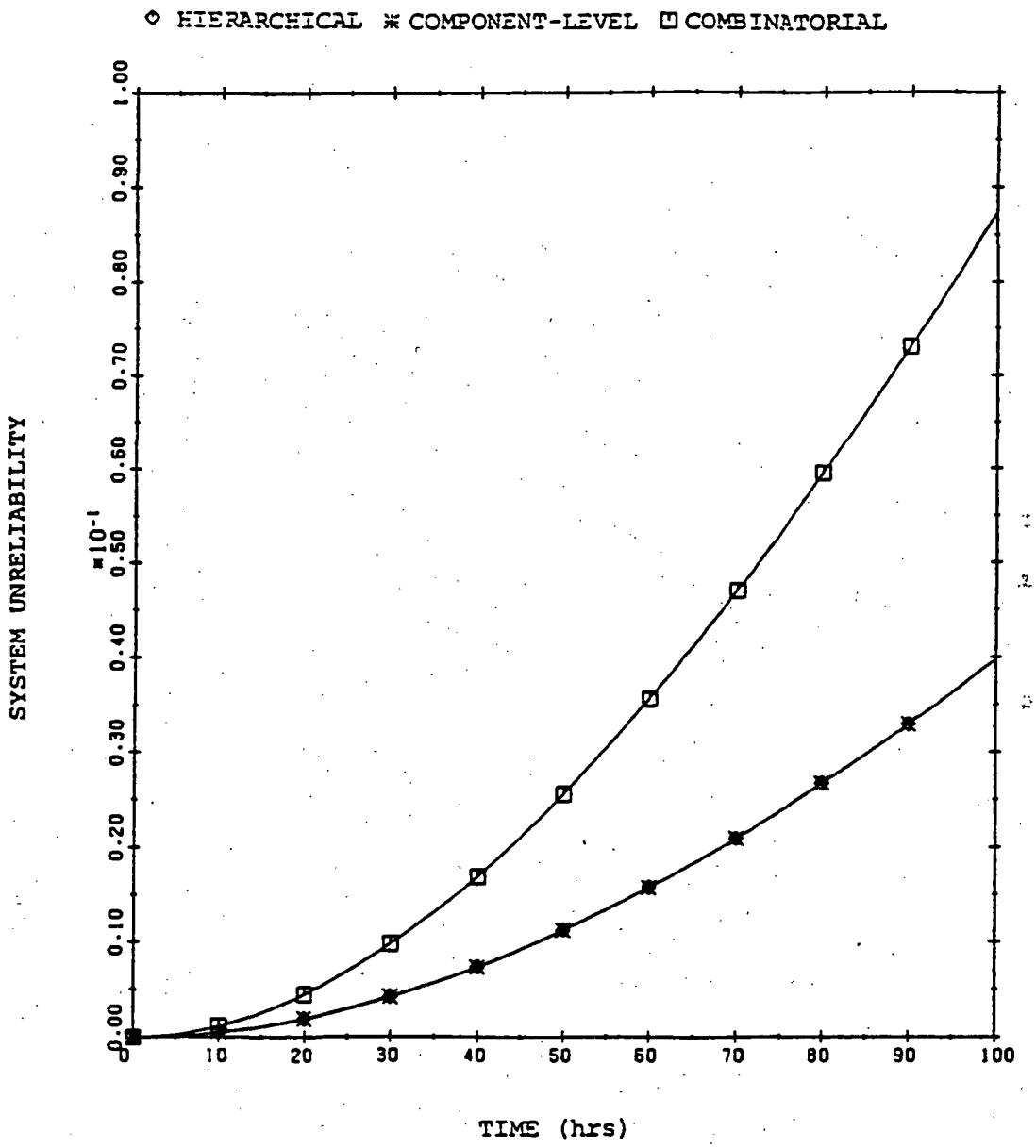


Figure 1.1
Motivating Example

sequence dependancy, and furthermore, produces state history probabilities for the system whereas the combinatorial approaches do not.

1.3 Methodology

The underlying concept of the hierarchical approach is that of aggregating component-level Markov model states into hierarchical states. The associated hierarchical state transitions are often time-varying to reflect multiple (non-simultaneous) failures within a given subsystem. These transitions correspond to subsystem-level events such as performance degradation and subsystem loss, and in turn usually imply system-level performance degradation. Consequently, the hierarchical model becomes semi-Markov because the holding time probability density functions for each hierarchical state are generally no longer the same for all transitions from a particular hierarchical state to other hierarchical states. In fact, the holding time distributions in the the hierarchical states are often Erlang (Ref{7}) (or more general) and this leads to a problem. For a semi-Markov chain where the failure rate from state i to state j can be expressed as $\lambda_{ij}(t)$ and $\lambda_i(t) = \sum_j \lambda_{ij}(t)$,

$$P_j(t) = P_j(0)e^{-\int_0^t \lambda_j(\tau)d\tau} + \sum_{i \neq j} \int_0^t P_i(x)\lambda_{ij}(x)e^{-\int_x^t \lambda_j(\tau)d\tau} dx \qquad (1.1)$$

(Ref{8}). Even for a small number of states this is a computational nightmare to solve. Note that the second term in equation 1.1 contains a convolution integral. This arises due to the dependence of a semi-Markov state's holding time on the time at which the state was entered (local time). The hierarchical approach exploits the Markov property and thus the local time dependence is memoryless. This makes the solution to the hierarchical model easier by assuming a single common holding time distribution for all exit transitions from a particular hierarchical state which allows us to write differential equations for the system state probabilities and eliminates the need to explicitly convolve.

In the hierarchical approach subsystem level events are addressed through the use of time-varying event rates computed from the component-level Markov models of the system's particular subsystems. These event rates are imbedded into the hierarchical system model. The resulting model is then evaluated over a mission time. Note that the subsystem events of interest are application-dependent because they depend on the system's redundancy management strategy. For example, one of the subsystem event rates might be associated with a particular degraded state while another event rate is the subsystem failure rate.

It is important to note a current restriction on the hierarchical approach. At this juncture the issue of repair has not been addressed

in the hierarchical approach. We also note that White, Butler and Lee (Ref{9},{10},{13}) examine a semi-Markov approach to unreliability evaluation. However, their approach does not address the state proliferation problem in that it utilizes component-level events rather than subsystem level events as in the hierarchical approach and thus does not reduce the state space of the fault-occurrence model.

1.4 Organization

The purpose of this thesis is to present a hierarchical approach to reliability modeling of fault-tolerant systems made up of fault-tolerant 'building blocks' (Ref{5}). The technique is developed in the context of an example architecture. The first section of Chapter Two characterizes the development of a hierarchical model. Section 2.2 defines the terms to be used in the derivation of the hierarchical approach and defines a hierarchical Markov state. Section 2.3 presents the sample architecture to which the hierarchical approach is applied. Section 2.4 gives a general form of the solution of a hierarchical model.

Chapter Three describes the implementation and simulation of the hierarchical approach as it is applied to the sample architecture of Section 2.3.

Chapter Four presents the results of the various test cases implemented.

Chapter Five summarizes the approach, the results of this study and presents topics for further research.

CHAPTER 2. A HIERARCHICAL APPROACH TO RELIABILITY MODELING

2.1 Outline of Hierarchical Model Development

Hierarchical model development is a four stage process as shown in figure 2.1. The process begins with a system definition. For a fault-tolerant system, a system definition involves defining the architecture and redundancy management strategy. From the system definition we identify the set of subsystem level events that are of interest. These include events which trigger reconfigurations and the reaching of various performance levels in a subsystem.

For the identified set of events, a set of component-level event models are developed. There will be an event model for each unique event defined. From each of these models, an (approximate) event rate can be derived. Using the event definitions and event rates, a hierarchical model of the system is developed. Each hierarchical state represents the system status with respect to the occurrence or non-occurrence of the set of subsystem events. The rates of occurrence of these events appear as the transition rates among the states of the hierarchical model.

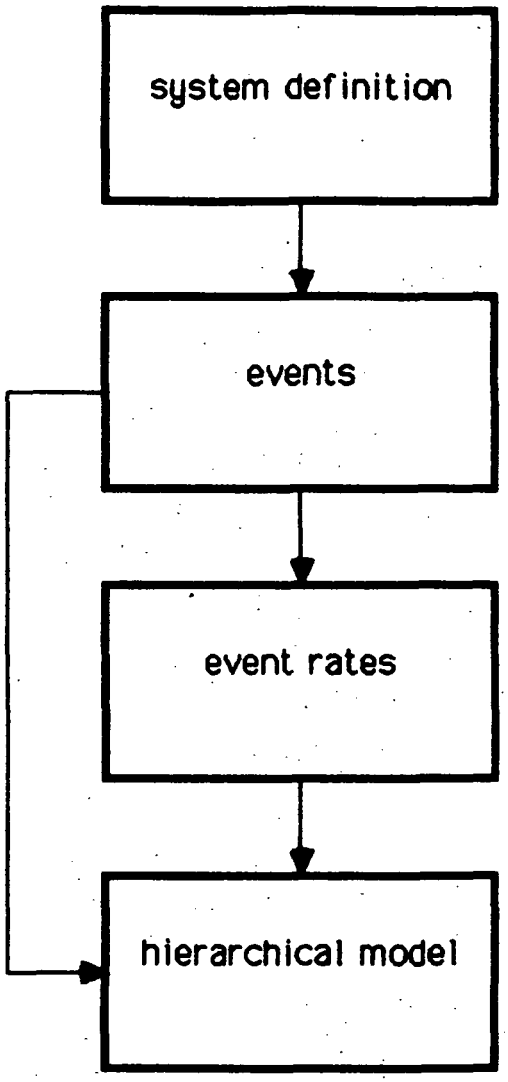


Figure 2.1
Hierarchical Model Development

The hierarchical model state descriptions are unique. We know that a full (no state aggregation or model truncation) unique component-level model of the system can be developed from the system definition. We also know that each state from that model maps uniquely into a hierarchical state. Thus the hierarchical model state descriptions are unique. Next we define the terms to be used in our development of the hierarchical approach.

2.2 Definitions

2.2.1 Definition of Terms

In deriving the hierarchical approach it is important to define the context and relationships between the terms used. Those terms are: component, subsystem, system, reconfiguration rules, component-level models, subsystem models, and hierarchical models. We begin with a simple example. Then a general form of the solution is presented for a more complex example which is used in the remainder of this work.

A simple example is used here to introduce the terms used in the hierarchical approach. The fault-tolerant system is made up of two fault-tolerant subsystems and is shown in figure 2.2.

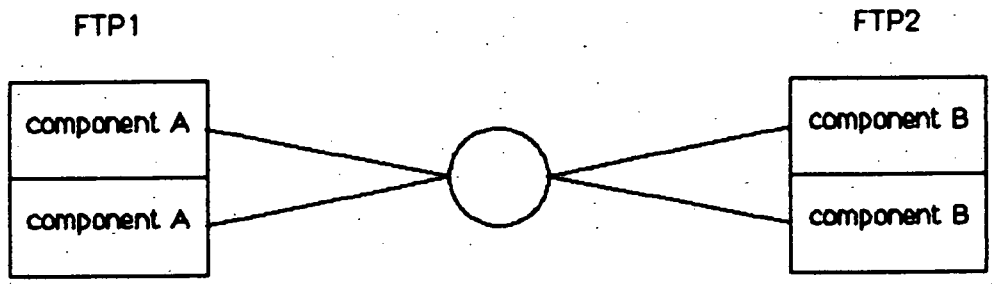
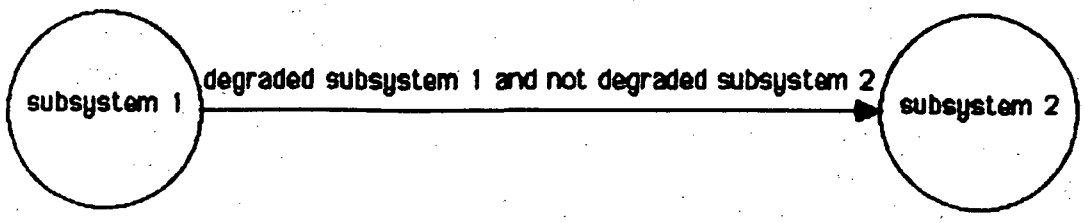


Figure 2.2
Simple-System Architecture

Each fault-tolerant subsystem is made up of two like components. However the two subsystems may have different components. A component is an element of a subsystem and often has a constant failure rate. Without a loss of generality (see Appendix A) we shall assume constant component failure rates throughout this study. Function migration means that system operation with respect to a particular function moves from one subsystem to the other subsystem according to the system reconfiguration rules. The system has the reconfiguration scheme given graphically in figure 2.3 and uses the following definitions.



Simple-System Reconfiguration Rules
Figure 2.3

A degraded subsystem is one that has one or more failed components. A failed subsystem is one that has two failed components.

Traditionally the component-level system model is formed as in figure 2.4.

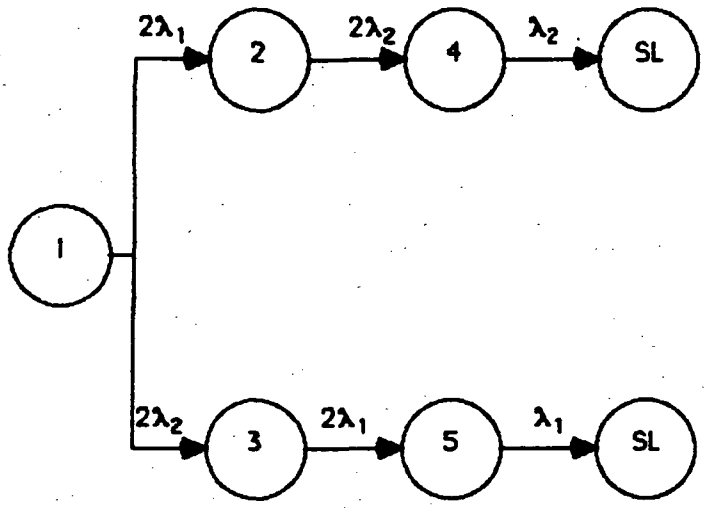


Figure 2.4
Simple-System Component-Level Model

For the hierarchical approach, component-level subsystem models are formed as in figure 2.5.

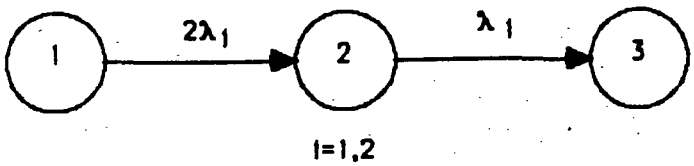


Figure 2.5
Component-Level Subsystem Model

From this component-level subsystem model, subsystem event rates are derived (this process is described in section 2.4). There are generally two types of subsystem event rates to be considered. These are: the rates at which a subsystem reaches various degraded states and the rate at which a subsystem becomes failed. Using such subsystem event rates, a hierarchical model of the system is formed (figure 2.6).

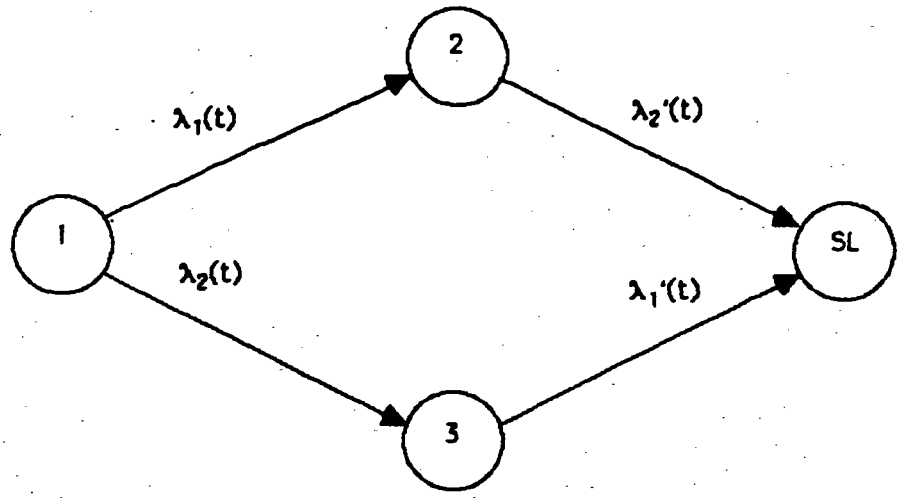


Figure 2.6
Simple-System Hierarchical Model

Comparing figures 2.4 and 2.6 we see that the hierarchical model aggregates component-level system model states.

Now that the essential elements have been introduced, a complete characterization of a hierarchical Markov state is given.

2.2.2 Definition of a Hierarchical Markov State

The essence of the hierarchical approach lies in the definition of a hierarchical Markov state. A hierarchical state captures the system status in terms of subsystem-level events rather than component-level events (as in the standard component-level Markov model approach Ref{4}). Normally, a Markov model generates the probability of having experienced a sequence of component failures. However, in the proposed approach, the hierarchical model generates the probability of having experienced a sequence of events which reflect the status of the system's subsystems. This is due to the fact that a hierarchical model state is an aggregation of those component-level system model states which have the same system status with respect to the pertinent subsystem-level events. Consequently, a hierarchical state probability is (approximately) the summation of the corresponding component-level system model state probabilities.

In order to derive the exit transition rates for the hierarchical states additional information must be associated with each hierarchical state. In particular, for each subsystem-level event corresponding to an exit transition from a given hierarchical state we must formulate an

event model which reflects both the time of entry into that particular hierarchical state and the possible states that the subsystem can be in whenthat hierarchical state is entered. The solution of the event model provides the time evolution of a conditional subsystem state probability vector (conditioned on being in a particular hierarchical state) which in turn provides the information necessary to derive the corresponding exit transition rate. The event models are readily derived from the component-level subsystem models as will be shown in Section 2.4. The derivation of the transition rates among hierarchical states will also be prescribed in Section 2.4. Presently we will describe the sample architecture used to illustrate these steps in the formulation of a hierarchical model. This example is also utilized to provide the numerical results of Chapter 4.

2.3 Sample Architecture

The sample architecture is a distributed processing system that consists of a collection of processing subsystems that are completely cross-strapped such that information can be exchanged between any processors. In formulating the hierarchical model in section 2.4, we will find that only two subsystems are needed in the sample architecture (figure 2.7) to fully describe the approach, its implementation, and associated problems. The ground rules for this system's operation prescribe that a quad fault-tolerant processor (FTP) (Ref{14}) is said to

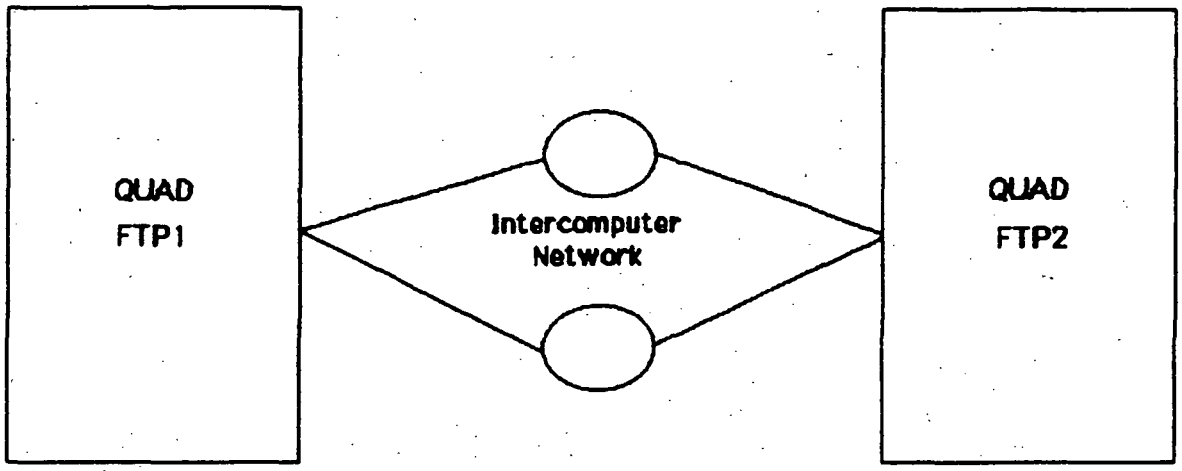


FIGURE 2.7
SAMPLE ARCHITECTURE

be operational if it has suffered no more than three channel (component) failures, and is said to be degraded (i.e., loses masking capability) if it has had two or more channel failures. A particular system function will remain in the initial supporting processor until that processor becomes degraded. The system will then migrate the function to a subsystem that is not yet degraded (if one is available), otherwise the function will remain in that subsystem until a third channel failure results in a system loss (see figure 2.8). In this example, the function of interest starts in FTP1 and migrates to the other subsystem as indicated in the hierarchical model in figure 2.9. Note that in this analysis, intercomputer network failures are neglected for the purpose of clarity. Note also that failure detection and isolation (FDI) of channel failures is assumed to be local to the subsystems. This is a necessary condition for the application of the hierarchical approach. However, coverage at the system level (i.e., probability of success of function migration) although not included in this example, can be readily incorporated in the state transitions of the hierarchical model. The derivation of the transition rates among the hierarchical states (see figure 2.9) is given in section 2.4.

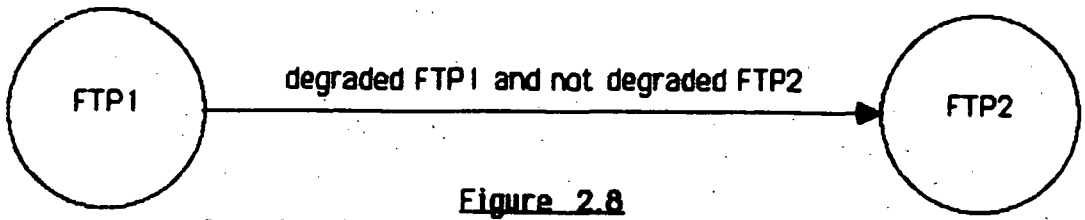


Figure 2.8
Sample Architecture Reconfiguration Rules

2.4 Hierarchical Model Development for Sample Architecture

For this sample architecture two different event models are required for each subsystem. This is because for each subsystem there are two distinct subsystem-level events intrinsic to the system definition. As previously stated, separate event models are needed to produce the con-

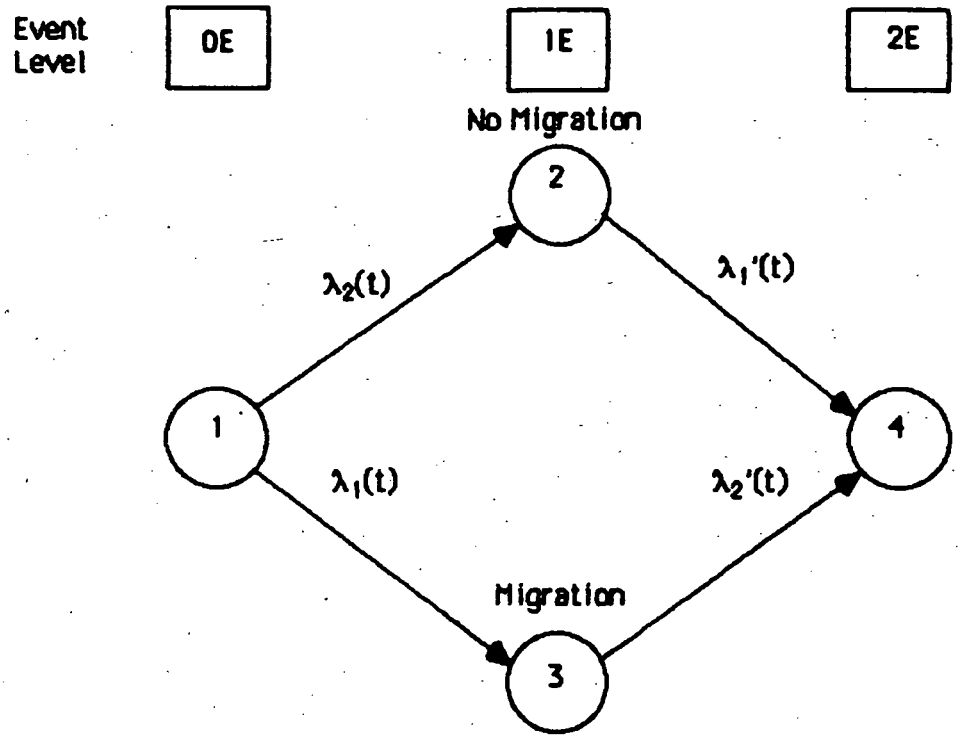


Figure 2.9
Hierarchical System Model

ditional subsystem state probability vectors associated with the calculation of hierarchical state transitions. These conditional subsystem state probability vectors obey the differential equation:

$$\dot{\pi}(t) = A\pi(t) + I_n u(t) \qquad (2.1)$$

Where $\Pi(t)$ is a conditional subsystem state probability vector associated with the hierarchical state for which exit transitions are to be derived. The $u(t)$ term is decomposed as follows:

$$u(t) = \text{pdf}\{\text{transition to the appropriate hierarchical state at time } t\}\rho(t) \qquad (2.2)$$

That is $u(t)$ is the joint probability which reflects the probability that the particular hierarchical state is transitioned into at some time t and the possible states that the subsystem can be in when that hierarchical state is entered. For example, if a function migrates to a subsystem at time τ, $u(\tau)$ will be:

$$u(\tau) = \text{pdf}(\tau)\rho(\tau) \qquad (2.3)$$

where $\rho_i(\tau)$ is the conditional probability (conditioned on the reconfiguration rules) that the functioning subsystem is in state i at time τ. $\rho(t)$ is a conditional subsystem state probability vector for the same subsystem as $\Pi(t)$ but reflects the possible states the subsystem can be in upon entry to the hierarchical state for which exit transition rates are to be derived. At this time is necessary to explain the pdf(τ) and $\rho(\tau)$ terms in detail.

2.4.1 Probability Density Function Determination

The pdf(τ) term represents the probability density function of the entry time to a particular hierarchical state. An isolated example is used to derive the pdf(τ) term in equation 2.3 and then the result is generalized to the hierarchical model given in figure 2.9. Given the simple four state hierarchical model in figure 2.10 (analogous but unrelated to the hierarchical system model given in figure 2.9), we need to know the probability density function of the time of transition to a particular state. Let the time of entrance to state i be the random variable τ_i. Given that the function of interest starts in state 0 with probability one, $\tau_0 = 0$ with certainty, ie.,

$$\text{pdf}(\tau_0) = \delta(0) \qquad (2.4)$$

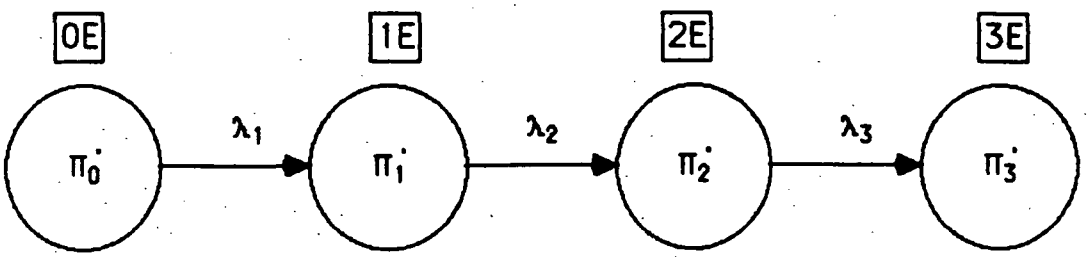


FIGURE 2.10
ISOLATED EXAMPLE

(a Dirac delta function at $\tau = 0$). Using the definitions of expected value and mean time to failure (MTTF) (Ref{11}), the pdf(τ_1) is derived. Defining the state 0 to state 1 transition as a subsystem failure, we have

$$\text{MTTF} = \int_0^{\infty} \tau_1 m(\tau_1) d\tau_1 \qquad (2.5)$$

where $m(\tau_1)$ is the mortality function:

$$m(\tau_1) = \lambda(\tau_1)R(\tau_1) \tag{2.6}$$

and, $R(\tau_1)$ is the reliability function,

$$R(\tau_1) = e^{-\int_0^{\tau_1} \lambda(t)dt} = \pi_0'(\tau_1) \tag{2.7}$$

(for this example). Thus,

$$m(\tau_1) = \lambda(\tau_1)\pi_0'(\tau_1) \tag{2.8}$$

and,

$$\text{MTTF} = \int_0^{\infty} \tau_1 \lambda(\tau_1)\pi_0'(\tau_1)d\tau_1 \tag{2.9}$$

And using the expected value result for a random variable X:

$$E\{X\} = \int X\,pdf(X)\,dX \qquad (2.10)$$

it can be seen by comparing equation 2.9 and equation 2.10 that:

$$pdf(\tau_1) = \lambda(\tau_1)\pi_0'(\tau_1) \qquad (2.11)$$

The pdf term is found in the same way for higher subsystem event levels, although it is not obvious that this is the case. Beyond the first subsystem event level the distribution of times to enter a hierarchical state would seem to become more involved computationally. Referring back to our isolated example in figure 2.10, the time of transition to state i is the random variable τ_1. Let the holding time in state i be a random variable h_1, (which depends on τ_1). Thus, to find the probability density function of time to enter state n (τ_n), an n-2 fold convolution is necessary. This is due to the fact that contained in the time to transition to state n are the holding times in all the states upstream of state n (which are all independent random variables). Because the holding times in the hierarchical states involve a cascade

of entry time dependencies, an exact determination of the density function of the time of transition to a hierarchical state at these higher event levels is computationally difficult. This difficulty increases rapidly with the event level of the hierarchical state in question.

The Markov property demands that the time spent in any state be "memoryless," thus a heavy constraint is imposed upon the distribution of times that a process can remain in a given state. To satisfy the Markov property, the holding time in a state of a continuous time process must be exponentially distributed with a parameter that depends only upon the state in question. That is, the holding time distribution is the solution to a first-order differential equation (Ref{7}). We have noted (see Section 1.3) that an exact formulation of the hierarchical model violates this restriction and is computationally difficult. Thus we must now introduce an appoximation.

If a hierarchical model is a semi-Markov process the holding time for a transition from some state i to any other state j may have an arbitrary distribution. We are imposing a restriction to force the hierarchical model to obey the Markov property. That is, in the hierarchical model we will assume that the holding time distribution for a transition from each state i to any other state j is exponential, satisfies the Markov property and is the same for any exiting transition. We hypothesize that this is a good approximation if all transitions out of each

hierarchical state have approximately the same holding time distributions.

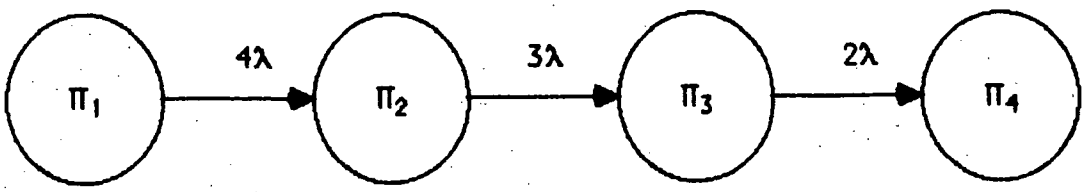


FIGURE 2.11
FTP1 MODEL

Due to the above approximation, the pdf term for an event model (equation 2.11) at any event level can easily be included in a difference equation which determines a subsystem's conditional state probability vector and does not contain a convolution to reflect the cascade of

local time dependencies. To start, we write the event models associated with a hierarchical state as differential equations of the form:

$$\dot{\pi}(t) = A\pi(t) + pdf(t)\rho(t) \qquad (2.12)$$

where A is the system matrix for the model shown in figure 2.11 and ρ is as given in equation 2.2. The pdf term is broken into its parts from equation 2.11:

$$pdf(t) = \sum_i \lambda_i(t)H_i(t) \qquad (2.13)$$

Where $H_i(t)$ is the probability of being in a hierarchical state with an exit transition into the hierarchical state associated with equation 2.12 and $\lambda_i(t)$ is the corresponding transition rate. Substituting equation 2.13 into equation 2.12, the exact solution to equation 2.12 is:

$$\pi(t) = e^{At}\pi(0) + \int_0^t e^{A(t-\tau)} \sum_i \lambda_i(\tau)H_i(\tau)\rho(\tau)d\tau \qquad (2.14)$$

which, when converted to discrete time is:

$$\pi(t_k) = \Phi(t_k,t_0)\pi(t_0) + \sum_{j=0}^{i=k-1} \Phi(t_k,t_j)pdf(t_j)\rho(t_j)\Delta t \quad (2.15)$$

where:

$$\Phi(t_k,t_0) = e^{A(t_k-t_0)} \quad (2.16)$$

Equation 2.15 is a convolution sum. However, it can be solved as a difference equation, stepping forward in time:

$$\pi(t_{k+1}) = \Phi(t_{k+1},t_k)\pi(t_k) + pdf(t_k)\rho(t_k)\Delta t \quad (2.17)$$

As mentioned earlier, the ρ term in the context of our example represents the conditional state probability vector for a subsystem when the function migrates there. The $pdf(t_k)\Delta t$ term represents the probability that the function migrates to that subsystem at time t_k. Both of these quantities are expressed in terms of the same global time argument.

Because both of these terms (pdf and ρ) are formulated in terms of the same time argument and are available at each time step, equation 2.17 can be updated at every time step. The hierarchical transition rates for the hierarchical model can then be constructed from $\Pi(t_k)$ as prescribed in Section 2.4.4.

2.4.2 Conditional Probability Derivation

The term $\rho(\tau)$ in $u(\tau)$ is a conditional subsystem state probability vector which reflects both the entry time to the hierarchical state associated with $\Pi(t)$ of equation 2.11 and the possible states that the subsystem can be in given the transition event and the associated source state. For the sample architecture we know that when a function migrates to a new processor, that the post-migration processor either in the zero channel failure state or the one channel failure state. And, given that all subsystems are active at system startup, there will be a unique conditional state probability (conditioned on having never entered states three of four) vector associated with the time of entry into a hierarchical state.

The Markov Model for a FTP subsystem is the same as that given in figure 2.11 and eq. 2.17. Thus, this model will have to be evaluated to produce the conditional subsystem state probability vector. The behavior of the model in figure 2.11 obeys the equation:

$$\dot{\pi}(t) = \begin{bmatrix} -4\lambda & 0 & 0 & 0 \\ 4\lambda & -3\lambda & 0 & 0 \\ 0 & 3\lambda & -2\lambda & 0 \\ 0 & 0 & 2\lambda & 0 \end{bmatrix} \pi(t) \qquad (2.18)$$

The conditional subsystem state probabiltiy vector $\rho(t)$ is computed by evaluating this subsystem state probability vector and normalizing it at each time step so that the subsystem can only be in states one and two. This is done as follows:

$$\rho_1(t) = \frac{\pi_1(t)}{\pi_1(t) + \pi_2(t)} \qquad (2.19)$$

and,

$$\rho_2(t) = \frac{\pi_2(t)}{\pi_1(t) + \pi_2(t)} \qquad (2.20)$$

for every time step. This gives the desired $\rho(t)$ for equation 2.12 and does so in a way that requires little computational overhead.

2.4.3 Initial Conditions

The initial conditions for each of the models associated with a hierarchical state are determined in a straight forward manner. Since system operation begins with no failures, the Markov subsystem models associated with hierarchical state one have the initial condition:

$$\pi_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad (2.21)$$

For subsystem models associated with the other hierarchical states, the initial conditions are:

$$\pi_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad (2.22)$$

2.4.4 Calculation of Subsystem Event Rates

The subsystem models are formulated as Markov chains in which states are reached via transitions which reflect component level events. A subsystem event rate must be computed for each of the events pertinent to a particular subsystem. To compute a subsystem event rate, a full Markov model of a subsystem (figure 2.11) is reduced to a simple two state Markov Model (figure 2.12) containing an operational state, a failed or degraded state (depending on the event in question) and a time-varying transition rate.

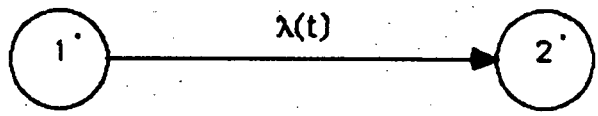


Figure 2.12
Transition Rate Example
Reduced Model

For the example in figure 2.11, state 4 is the failure state and states 1-3 are the operational states. The transition rate in figure 2.12 is computed as follows:

$$\lambda(t) = \sum_{i \in \{\text{operational states}\}} \lambda_{i4} P\{\text{in state } i \text{ at } t \mid \text{system occupies an operational state at } t\} \qquad (2.23)$$

So, for this example,

$$\lambda(t) = \frac{\lambda_{14}\pi_1(t) + \lambda_{24}\pi_2(t) + \lambda_{34}\pi_3(t)}{\pi_1(t) + \pi_2(t) + \pi_3(t)} \qquad (2.24)$$

however, λ_{14} and λ_{24} represent the rates of simultaneous failure events, which are negligible. Thus, for the given example,

$$\lambda(t) = \frac{\lambda_{34}\pi_3}{1 - \pi_4(t)} \qquad (2.25)$$

is the rate of the event of subsystem failure which corresponds to $\lambda_1'(t)$ and $\lambda_2'(t)$ in Figure 2.6. The subsystem degradation rates $\lambda_1(t)$ and $\lambda_2(t)$ of Figure 2.6 are obtained in a similar manner.

Another factor in determining the accuracy of a hierarchical model is the ability of the derived event rate to accurately model the dynamics of the component-level system model states that are aggregated into the source state for the rate in question.

CHAPTER 3. IMPLEMENTATION AND SIMULATION

Unreliability predictions for the sample architecture are obtained in two ways: a component-level Markov chain (exact model) and the hierarchical procedure described in Chapter 2. The event models associated with the hierarchical approach are also component-level Markov chains and because we have assumed constant component failure rates (see Appendix A) these models and the exact model can be readily solved via a discrete-time time-invariant state transition matrix (STM) formulation. In the case of the hierarchical model time varying transition rates appear and consequently this model must be numerically integrated. Both methods are described below.

3.1 Simulation Implementation

3.1.1 Solution of Event and Exact Models

Since the event and exact models are linear time-invariant Markov chains, a discrete-time time-invariant state transition matrix formulation can be used to propagate the state probability vectors through time. The general system to be solved is:

$$\dot{x}(t) = Ax(t) + Bu(t) \qquad (3.1)$$

Because the homogeneous solution is a degenerate case of the complete solution of the state equation (equation 3.1), that result will be shown later. The solution of equation 3.1 is:

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau \qquad (3.2)$$

The terms e^{At} and $e^{A(t-\tau)}$ are the STM for the system in equation 3.1. Putting equation 3.2 in terms of a STM Φ :

$$x(t) = \Phi(t - t_0)x(t_0) + \int_{t_0}^t \Phi(t - \tau)Bu(\tau)d\tau \qquad t > t_0 \qquad (3.3)$$

To obtain a discrete-time representation of equation 3.3 a constant time step, Δt, is needed. Defining

$$\Delta t = t_k - t_{k-1} \qquad k = 1,2,... \qquad (3.4)$$

and,

$$t_k = k\Delta t \tag{3.5}$$

with,

$$t_0 = (k - 1)\Delta t \qquad k = 1 \tag{3.6}$$

Assuming u(t) is constant over each time step, equation 3.2 becomes:

$$x(t_k) = e^{A\Delta t}x(t_{k-1}) + \left(\int_0^{\Delta t} e^{A\tau}d\tau\right)Bu(t_{k-1}) \tag{3.7}$$

Using the single step STM $\Phi_{ss} = e^{A\Delta t}$, equation 3.7 becomes:

$$x(t_k) = \Phi_{ss}(\Delta t)x(t_{k-1}) + \left(\int_0^{\Delta t} e^{A\tau}d\tau\right)Bu(t_{k-1}) \tag{3.8}$$

Given that B is a constant matrix,

$$x(t_k) = \Phi_{ss}(\Delta t)x(t_{k-1}) + Bu(t_{k-1})\Delta t \qquad (3.9)$$

The single step STM must be put in terms that can easily be computed. We know that:

$$\Phi_{ss}(\Delta t) = e^{A\Delta t} \qquad (3.10)$$

which expands to:

$$e^{A\Delta t} = \sum_{i=0}^{\infty} \frac{(A\Delta t)^i}{i!} \qquad (3.11)$$

For Δt much smaller than the reciprocal of the largest magnitude eigenvalue of A, the expansion can be truncated after a few terms and still retain the desired accuracy (Ref {2}). So for this implementation, the single step STM is approximated as:

$$\Phi_{ss}(\Delta t) = I + A\Delta t \tag{3.12}$$

For the purely homogenous cases (i.e. the exact model and event models associated with the first hierarchical state), $u(t) = 0$ and the solution of the state equation becomes:

$$x(t_k) = \Phi_{ss}(\Delta t)x(t_{k-1}) \tag{3.13}$$

For the subsystem models of FTP1 and FTP2 in the sample architecture,

$$\Phi_{ss}(\Delta t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} -4\lambda & 0 & 0 & 0 \\ 4\lambda & -3\lambda & 0 & 0 \\ 0 & 3\lambda & -2\lambda & 0 \\ 0 & 0 & 2\lambda & 0 \end{bmatrix} \Delta t \tag{3.14}$$

For the exact model, the single step STM is (see Figure 3.1):

$$\Phi_{ss}(\Delta t) = I_{11} + A_E \Delta t \qquad (3.15a)$$

$$A_E = \left[\begin{array}{c} \text{See next page} \end{array}\right] \qquad (3.15b)$$

The homogenous models are solved simply by stepping forward in time using equation 3.13.

For determining the conditional subsystem state probability vectors associated with the hierarchical states at the first and greater event levels, the B matrix is the identity matrix and $u(t_k)$ is as given in equation 2.3. So again the model is solved simply by stepping forward in time using equation 3.9.

3.1.2 Solution of Hierarchical Model

As formulated, the hierarchical model is simply a first order, homogeneous, linear system of differential equations with time-varying coefficients. The general form of this system is:

$$\dot{x}(t) = A(t)x(t) \qquad (3.16)$$

$$
\begin{bmatrix}
-4(\lambda_1+\lambda_2) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
4\lambda_1 & -3\lambda_1-4\lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
4\lambda_2 & 0 & -4\lambda_1-3\lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 3\lambda_1 & 0 & -4\lambda_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 4\lambda_2 & 4\lambda_1 & 0 & -3(\lambda_1+\lambda_2) & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 3\lambda_2 & 0 & 0 & -4\lambda_1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 4\lambda_2 & 3\lambda_1 & 0 & -3\lambda_2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 3\lambda_2 & 4\lambda_1 & 0 & -3\lambda_1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 3\lambda_2 & 0 & -2\lambda_2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 3\lambda_1 & 0 & -2\lambda_1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\lambda_2 & 2\lambda_1 & 0
\end{bmatrix}
\tag{3.15b}
$$

Component-Level Exact Model Matrix

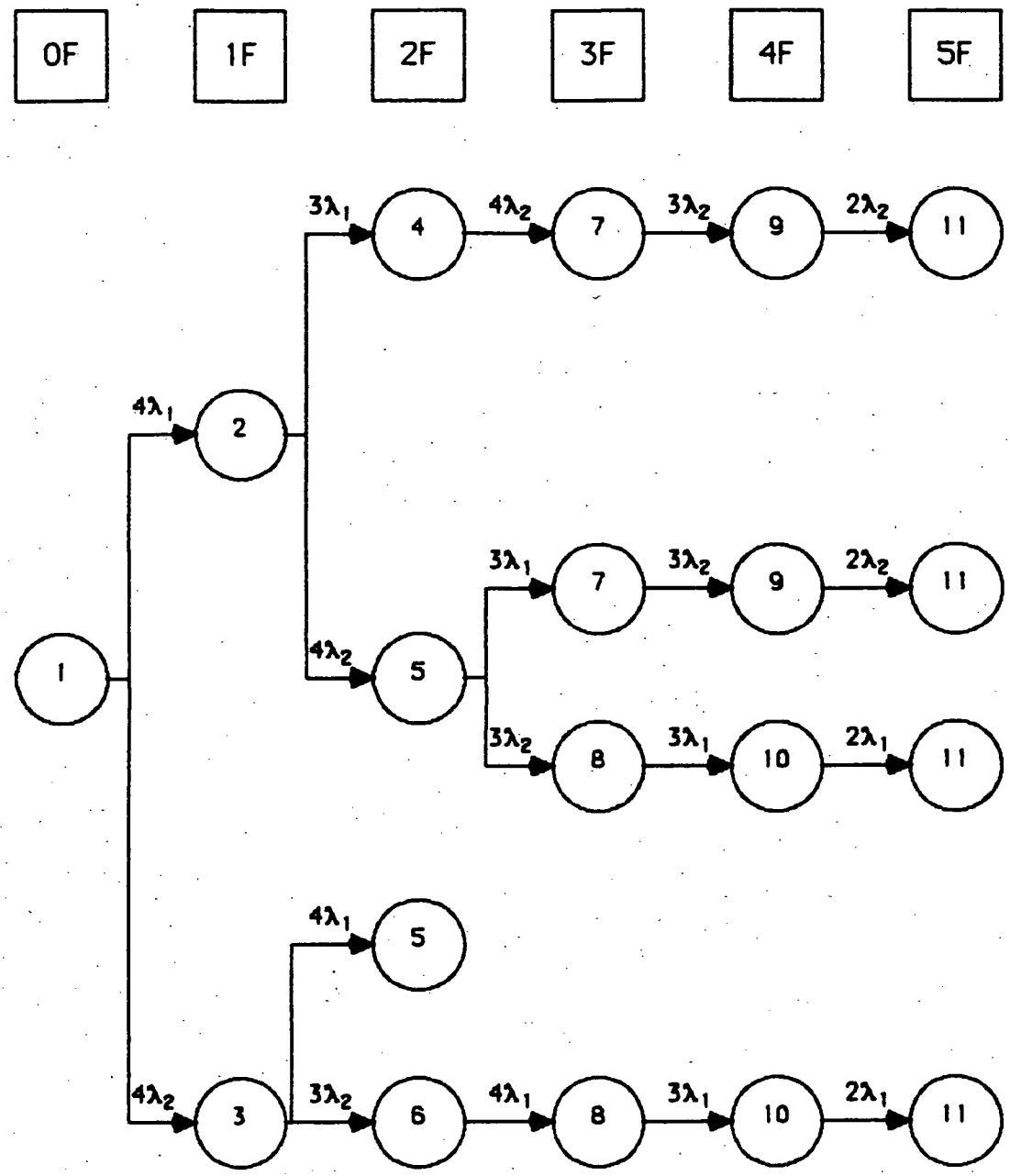


Figure 3.1
Component-Level Model
Two Subsystems

This is a simple initial value problem that must be integrated to produce the result.

There are generally two ways in which an initial value problem can be solved: a single-step method (Euler method, Runge-Kutta method) and multistep methods (Predictor-Corrector methods) (Ref {12}). The choice consists of a trade-off between accuracy and computational efficiency. The method used in this study is a 5 and 6 stage Runge-Kutta method. This was chosen due to the moderate relative error requirement based on the small number of significant digits (1 or 2 at best) in the input and consequently the solution of the truth model. An outline of the Runge-Kutta method follows:

1. Take a step Δt forward from t using the Euler method.
2. Evaluate $x(t)$ at this point and use $x(t+\Delta t)$ to adjust derivative to be used at t.
3. Use adjusted slope to take a second step from t.
4. Evaluate $x(t)$ at this point and further adjust slope to be used at t.
5. Repeat 3 and 4 to the order desired.
6. Combine all estimates to take actual step to $t+\Delta t$

It is pointless to rederive the Runge-Kutta formulas here. The reader is directed to Rice for the details of the formulas.

The actual subroutine used here is DVERK from the IMSL package. It was chosen because it solved the hierarchical models to close agreement with the exact models. If we loosen our relative error requirement, a less computationally expensive method (such as the single step, single stage Euler Method) could be used and introduce even greater computational savings through the use of the hierarchical approach.

3.1.3 Program Outline

The hierarchical model and event models are solved simultaneously as the event models provide information needed to construct the time-varying transition rates for the hierarchical model.The exact model is also solved simultaneously so that the hierarchical results can easily be compared to the exact model. An outline of the simulation program follows:

- Read initial data
- Initialize subsystem and hierarchical state probability vectors
- Initialize exact model state probability vector
- Assemble event model STMs
- Assemble exact model STM
- For t_0 to $t_{mission}$ by Δt
    - Update exact model
    - Compute hierarchical transition rates from event model results
    - Assemble hierarchical system matrix

- Integrate hierarchical system state equations
- Output time,hierarchical unreliability prediction, exact model unreliability prediction
- Update conditional subsystem state probability vectors

- Output run statistics

CHAPTER 4. RESULTS

4.1 Computational Efficiency

Probably the most important result of the hierarchical modeling technique is the increased computational efficiency compared to the component-level modeling approach. When a large number of subsystems exist in a system and many combinations of reconfigurations are of interest, the state-space of a component level model grows rapidly with the number and complexity of subsystems. In contrast, the hierarchical approach does not add nearly as many states to the system-level model when another subsystem is added. Consequently there is potentially a considerable difference in the computational burden associated with each approach. To demonstrate this observation we will examine the hierarchical and exact models for three systems comprised of different subsystems. The three cases examined are: two dissimilar subsystems, two dissimilar subsystems with imperfect coverage at the subsystem level, and three dissimilar subsystems.

We shall count the number of multiplications needed per time step. The assumption is made that, for the purpose of comparing computational efficiency, the hierarchical model and the exact model are solved using the same numerical technique. Note that in formulating the exact mod-

els, considerable state reduction was performed via state aggregation based upon common exit transition rates. In addition we exploited the observation that once a FTP loses two channels, additional channel failures need not be tracked since that FTP is no longer a candidate post-migration site; the only exception is the case of the final supporting FTP where channel failures must be tracked to the point of total system failure. Also note that the operation of multiplying a N-vector by a NxN matrix is of order N^2.

For the case of two dissimilar subsystems each subsystem model is fourth order (see Figure 2.11) and the hierarchical model is fourth order. Counting multiplications per time step we have:

model	multiplications/Δt
0th level event 1st level event hierarchical	$2x3^2$ $2x(4^2+2)$ 4^2
sum	70

Table 4.1

Notice that the two event models associated with the 0th event level have only three states. This follows from the observation that we are only interested in the transition between states 2 and 3 in Figure 2.11 as this transition corresponds to the event that triggers the function migration. This observation will be utilized in the other two cases. The component-level exact model for this case has the following order of execution:

model	multiplications/Δt
exact	11^2
sum	121

Table 4.2

The next two cases show, when compared to the first, that a hierarchical model's computational advantage over a component-level model of the system increases as the subsystem models become more complex and also as more subsystems are added to the system. To address the issue of more complex subsystem models, we will no longer assume perfect coverage at the subsystem level. Instead we will include a vulnerable

state in the subsystem model to capture near-coincident failures in a subsystem which will lead to the loss of that subsystem. Such near coincident failures arise when an adjudication is to be made on the occurence of a failure among three components. If two components fail coincidently, the ability to correctly isolate the failed components may be lost. Consequently the vulnerable state is included between states 2 and 3 of Figure 2.11. This state has two exit transitions one of which reflects the automatic failure detection, isolation and reconfiguration (FDIR) rate and the other the occurance of a second channel failure which yields a subsystem failure. The addition of the vulnerable state makes the subsystem models fifth order. Counting multiplications per time step in the hierarchical model we have:

model	multiplications/Δt
0th level event 1st level event hierarchical	$2x4^2$ $2x(5^2+2)$ 4^2
sum	102

Table 4.3

Note that the 0th level event model is reduced in order as before. The component-level exact model for this system has the following order of execution:

model	multiplications/Δt
exact	18^2
sum	324

Table 4.4

Our third case addresses the inclusion of additional subsystems in the system. The system now includes three dissimilar subsystems and once again we assume perfect coverage in the subsystems. The event models are again fourth order, but the hierarchical model is now eighth order. Counting multiplications per time step we have:

model	multiplications/Δt
0th level event 1st level event 2nd level event hierarchical	$3x3^2$ $6x(4^2+2)$ $3x(4^2+2)$ 8^2
sum	253

Table 4.5

The component-level exact models for this case has 30 states and the following order of execution:

model	multiplications/Δt
exact	30^2
sum	900

Table 4.6

The above results clearly demonstrate the savings in computation when the hierarchical approach is utilized. The worst case for the hierar-

chical approach from a computational viewpoint occurs when all the subsystems are different. Then, there must be a distinct event model for each state transition of the hierarchical model. Even in this case, however, the hierarchical approach requires less computation than a component-level model for non-trivial systems. For systems wherein some (or all) subsystems are identical the number of event models required is less than the number of state transitions appearing in a full (i.e. no state reduction performed) hierarchical model. The number of event models required is equal to the number of state transitions remaining after the order of the full hierarchical model has been reduced by performing state aggregation on the basis of common exit transition rates.

4.2 Two Subsystem Results

As mentioned previously, the two subsystem architecture fully describes the hierarchical approach and associated problems. Thus, all test cases utilized the models developed for the sample architecture described in Chapter 2. Two sets of cases are presented: similar subsystems, and dissimilar subsystems.

4.2.1 Similar Subsystems

The parameter to be varied in this section is the ratio of component MTBF (mean-time-between-failure) to mission time. All plots are compar-

isons of the unreliability predictions of the hierarchical and of the component-level approaches for the given system versus time. Figure 4.1 shows the results for a MTBF to mission time ratio of one. Figure 4.2 gives the results for a MTBF to mission time ratio of five. Figure 4.3 is the result for a MTBF to mission time ratio of ten. We see that all plots show good agreement between the two approaches. Figure 4.4 shows the absolute value of the percentage of relative error between the hierarchical model's and exact model's unreliability prediction given in figure 4.3. We see that there is indeed an approximation introduced utilizing the hierarchical model as the percentage of relative error is larger than expected errors due to machine precision. Note that the large relative error early in the mission is a result of the reduced observability for the first few time steps in the hierarchical model. The error plots for the previous two cases demonstrate similar behavior and therefore are not included.

4.2.2 Dissimilar Subsystems

In this section, the component MTBF to mission time ratio are different for the two subsystems. In the first case (figure 4.5) subsystem 1 has an MTBF to mission time ratio of ten. In this case for subsystem 2

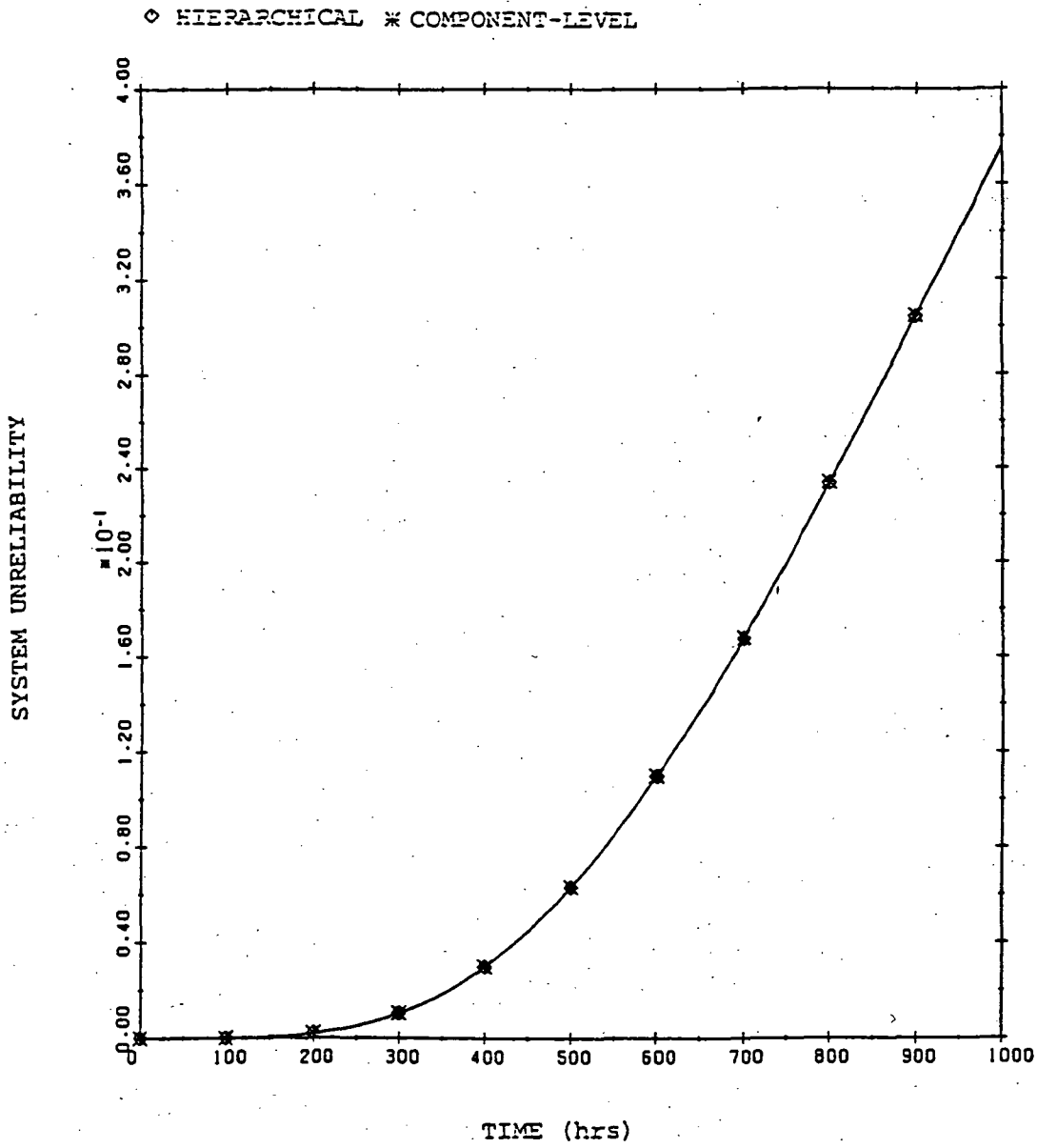


Figure 4.1
Identical Subsystems
component MTBF:mission time = 1.0

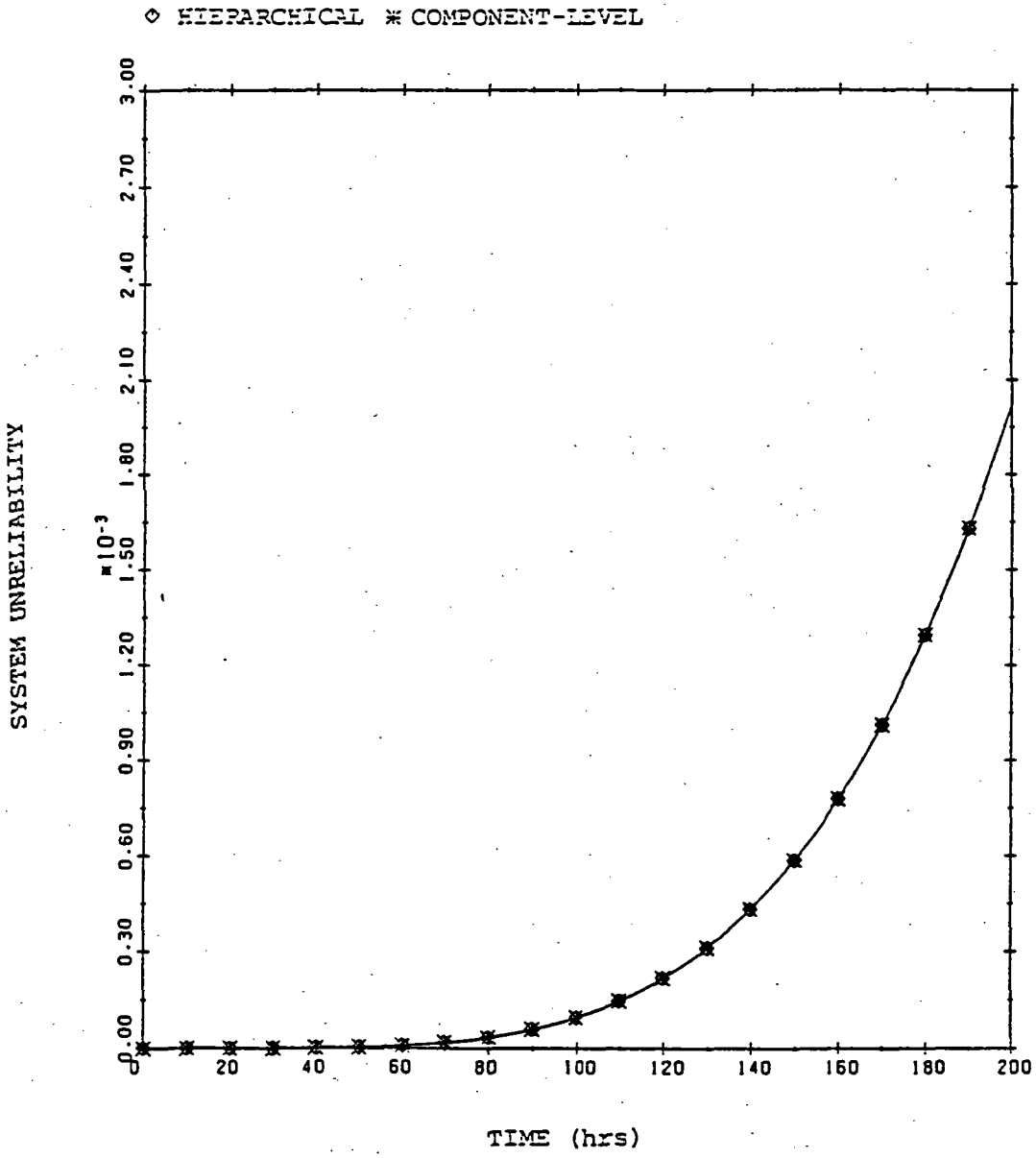


Figure 4.2
Identical Subsystems
component MTBF:mission time = 5.0

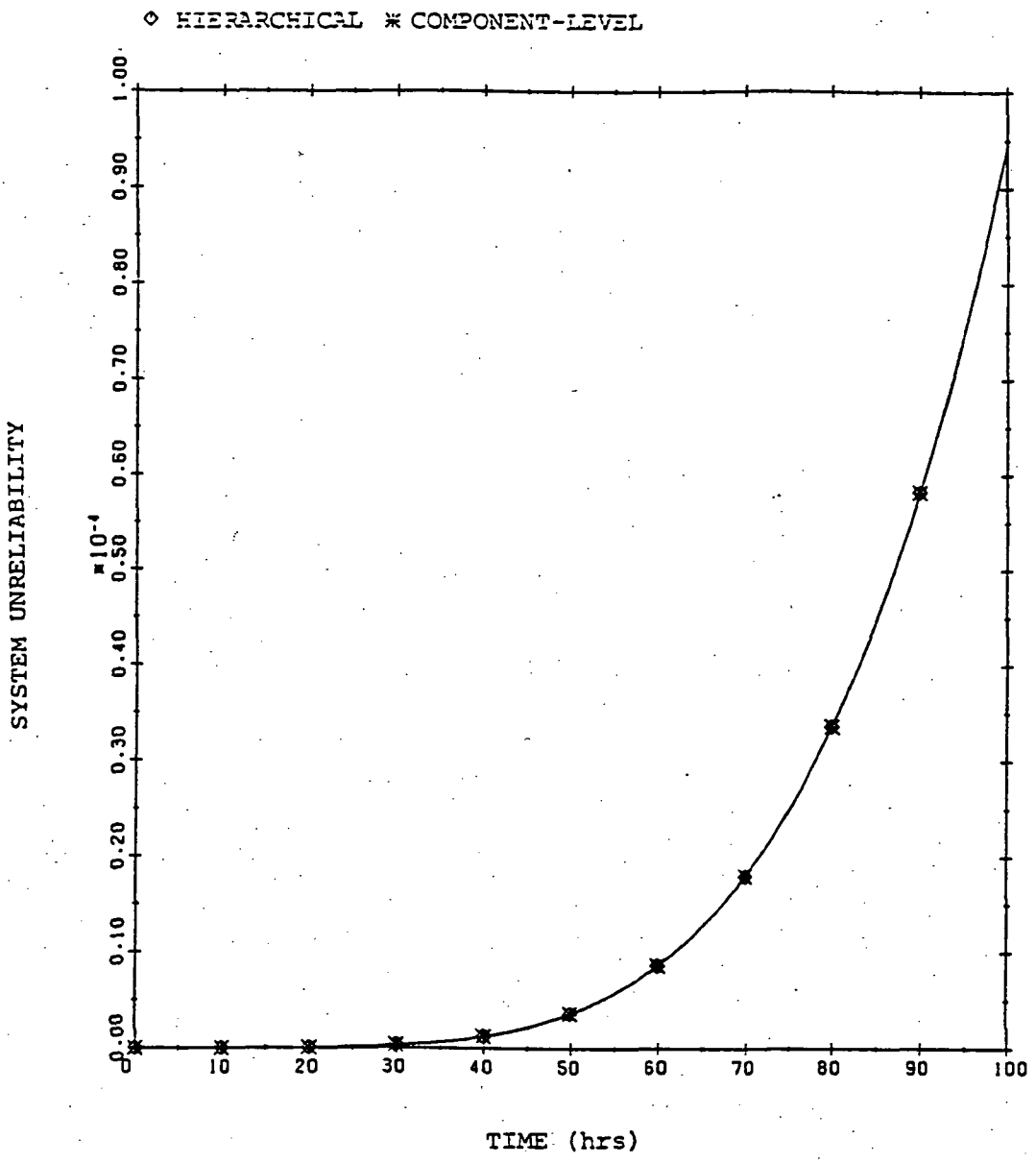


Figure 4.3
Identical Subsystems
component MTBF:mission time = 10.0

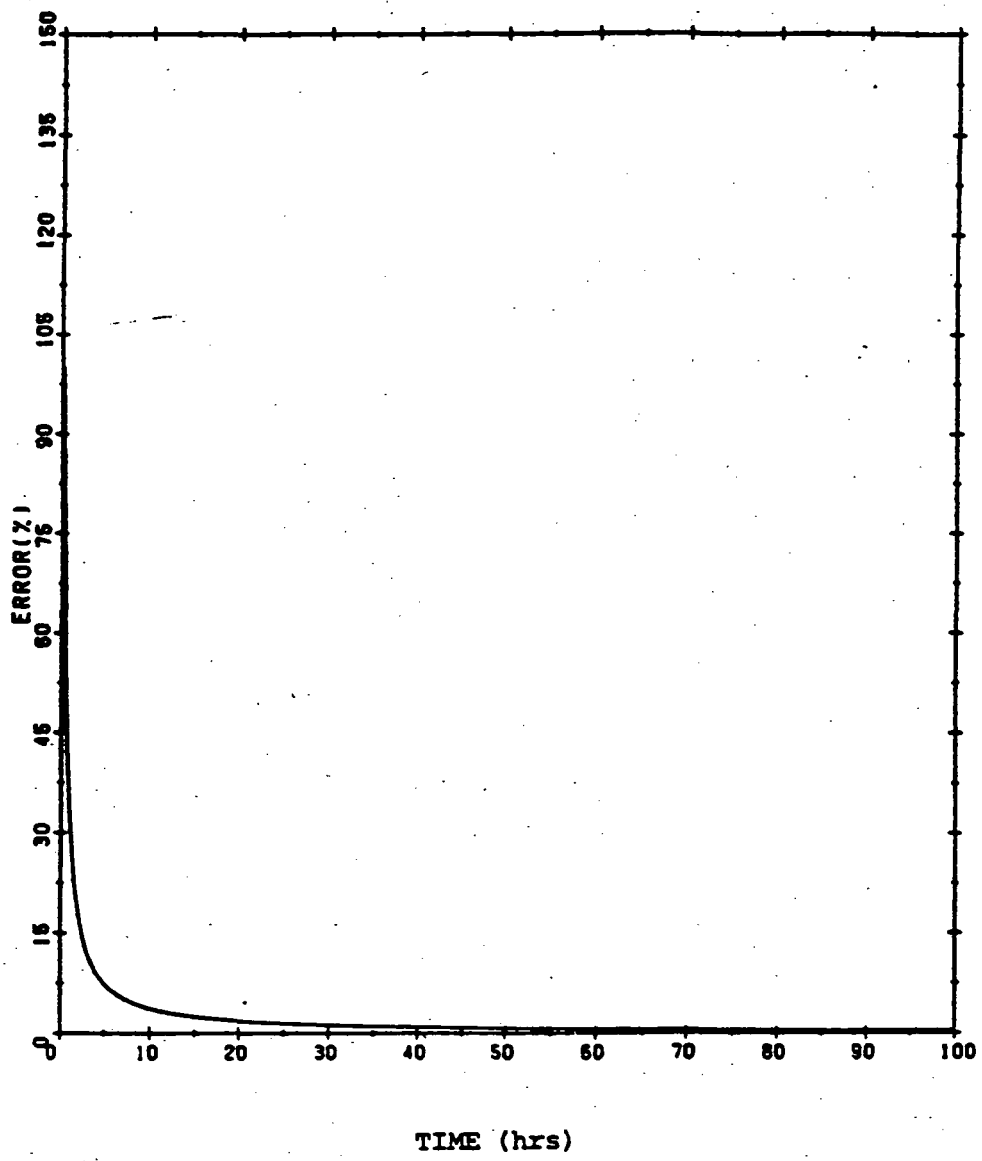


Figure 4.4
Relative Error Percentage
Identical Subsystems
component MTBF:mission time = 10.0

this ratio is equal to one. In figure 4.6 subsystem 1 has an MTBF to mission time ratio of 10 while this ratio for subsystem 2 is equal to .1. Again the plots show the predictions of system unreliability by the hierarchical approach and by the component-level exact model against time.

We see that there is disagreement between the curves in both figure 4.5 and figure 4.6. Note that the amount of disagreement is larger for the smaller difference in component failure rate between the subsystems (figure 4.5). Also note that the hierarchical approach is neither consistently conservative nor consistently optimistic in its prediction of system unreliability. Figures 4.7 and 4.8 give the absolute value of the percentage of relative error between the hierarchical model's and exact model's unreliability predictions given in Figures 4.5 and 4.6 respectively. Again we see that the approximation introduced using the hierarchical model is larger than can be attributed to computer round-off error.

Figure 4.3 has the same mission time as figures 4.5 and 4.6 but has identical component failure rates in the subsystems. In this case, there is excellent agreement with the exact model. So, the trend we see in figures 4.3, 4.5 and 4.6 is that as the difference in the component failure rates between the subsystems is increased, the error in the

hierarchical approach increases, and then decreases for the example given.

In Chapter Two we said the validity of the hierarchical approach depends on whether all transitions out of a hierarchical state to any other hierarchical state are first-order and the same for any transition. In light of this, the hierarchical method is expected to give very small error for identical subsystems. If the subsystems are identical, $\lambda_2(t)$ equals $\lambda_1(t)$ in figure 2.9. As the component failure rates are varied, our assumption breaks down. However, when one subsystem's component failure rate is much larger than the other subsystem's, the exit transitions from hierarchical state 1 in figure 2.9 become essentially a single transition rate. Put another way, a dominant failure mode surfaces and the other mode becomes negligible in the computation of unreliability by both the hierarchical approach and component-level exact model. So we see that the hierarchical approach is not always conservative, but the associated error is small for the cases studied.

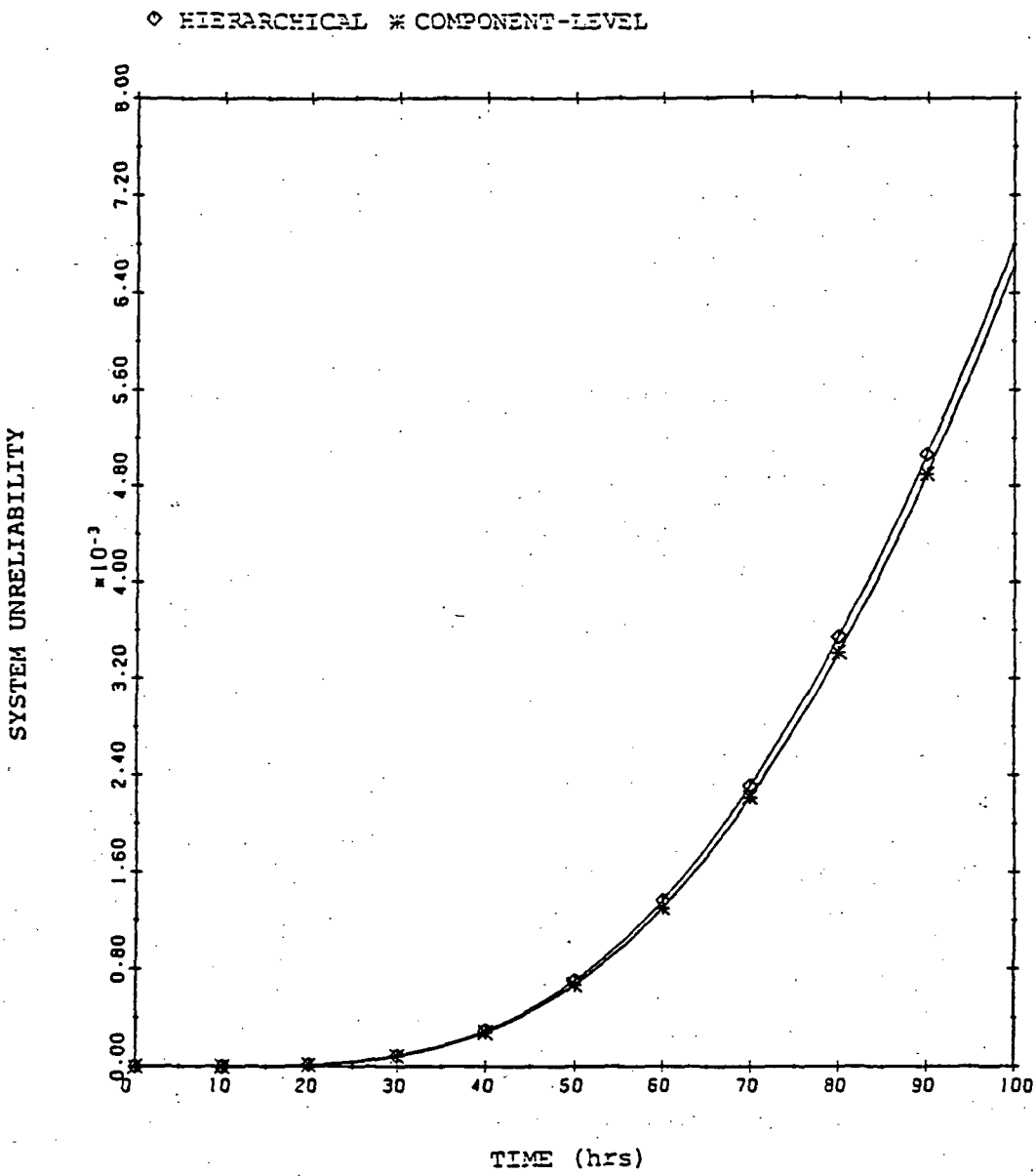


Figure 4.5
Different Subsystems
subsystem 1: component MTBF:mission time = 10.0
subsystem 2: component MTBF:mission time = 1.0

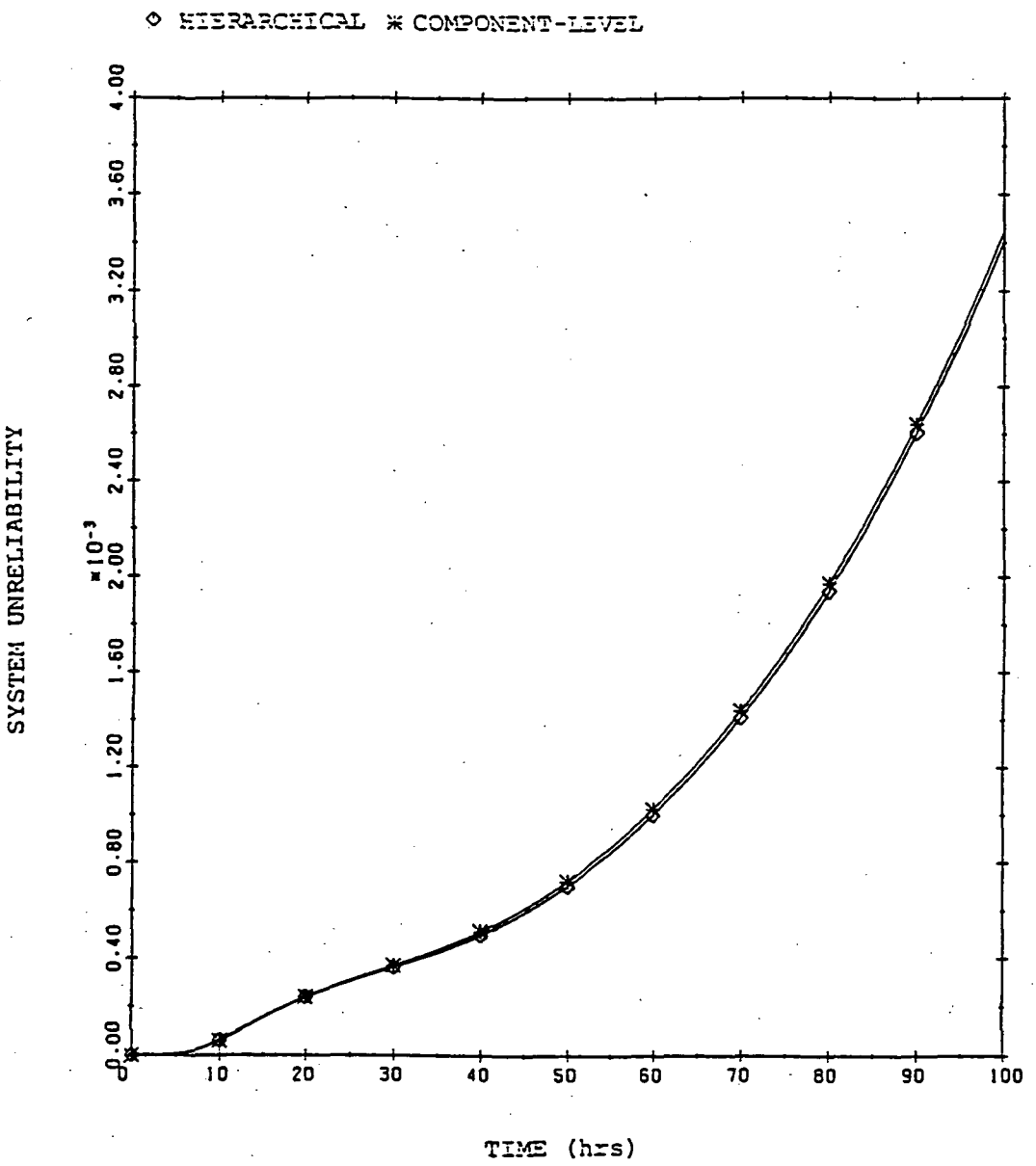


Figure 4.6
Different Subsystems
subsystem 1: component MTBF:mission time = 10.0
subsystem 2: component MTBF: mission time = 0.1

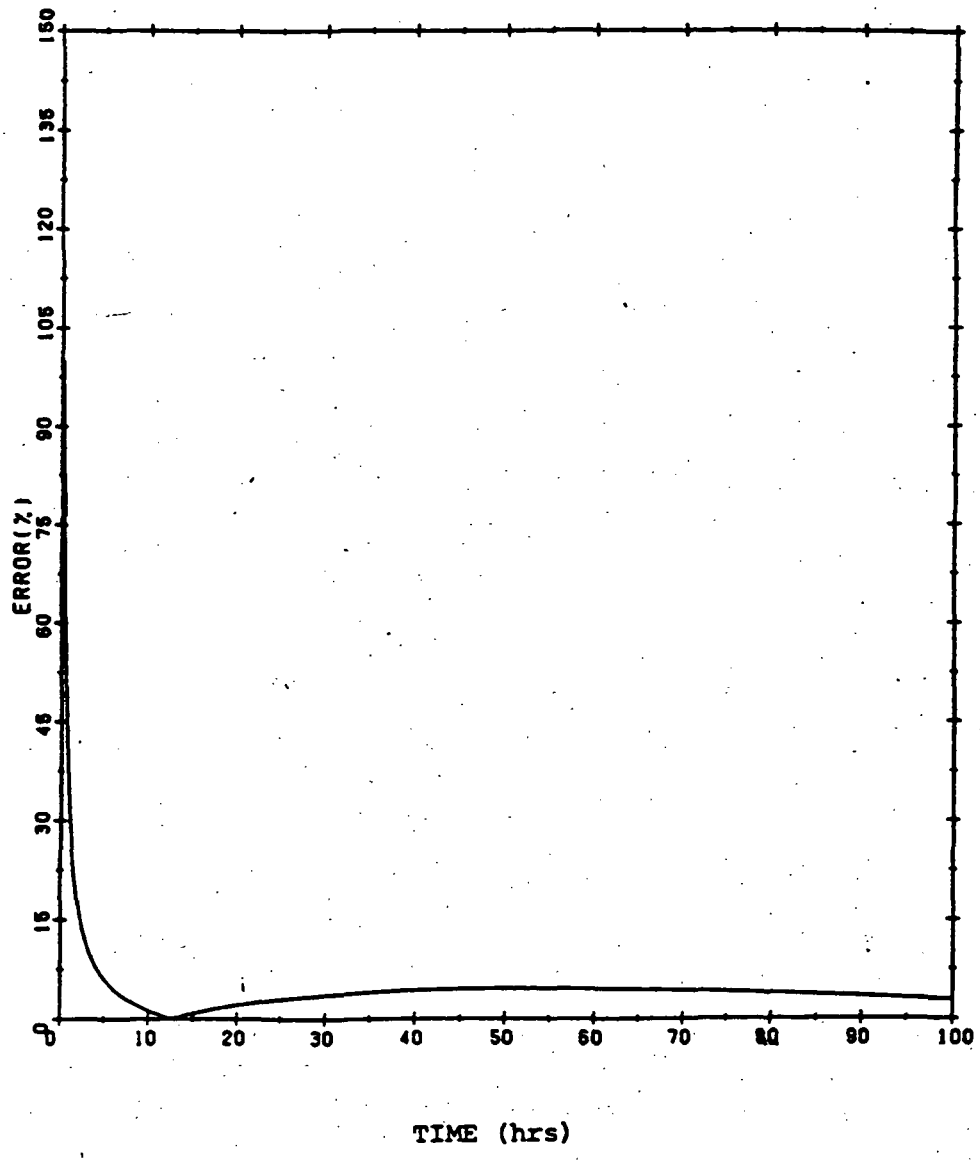


Figure 4.7
Relative Error Percentage
subsystem 1: component MTBF:mission time = 10.0
subsystem 2: component MTBF:mission time = 1.0

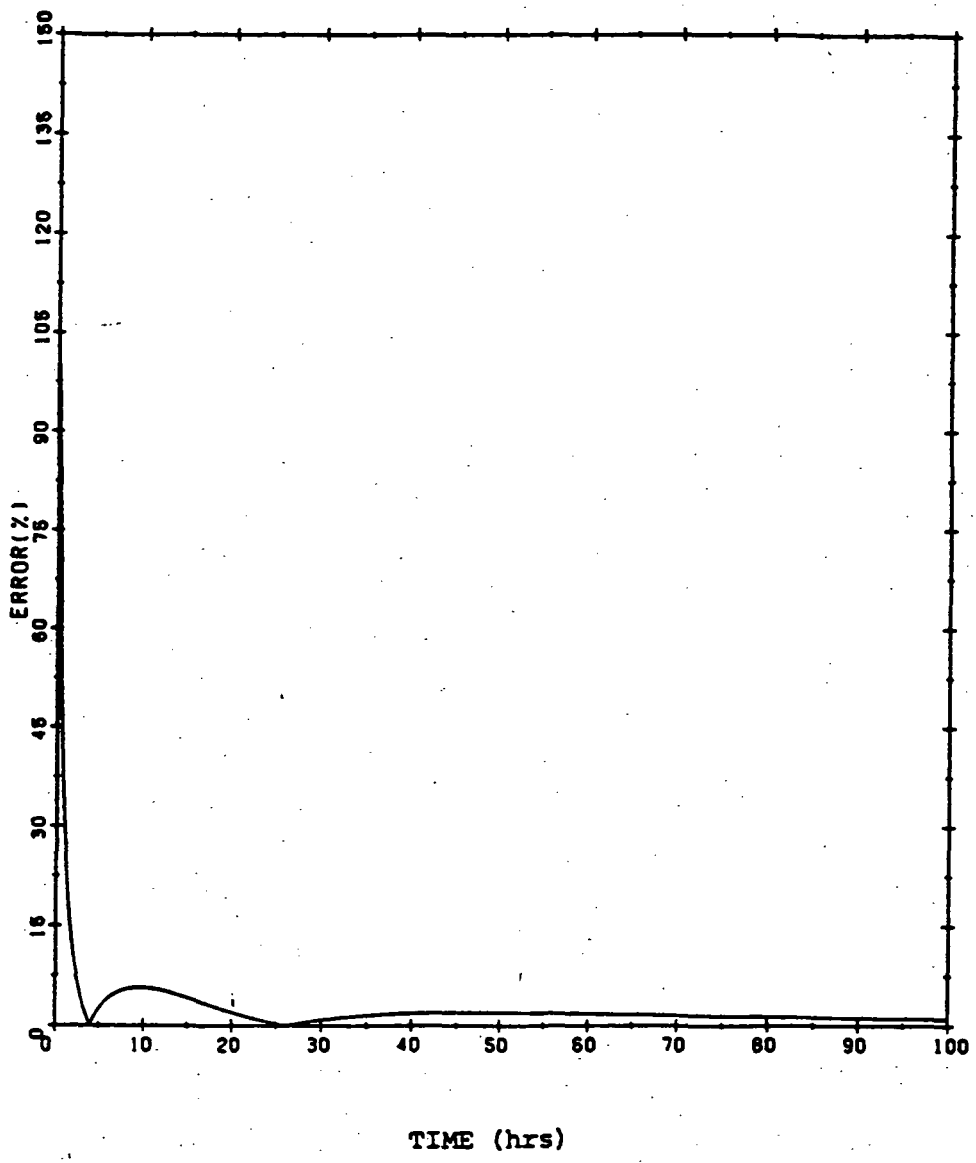


Figure 4.8
Relative Error Percentage
subsystem 1: component MTBF:mission time = 10.0
subsystem 2: component MTBF:mission time = 0.1

CHAPTER 5. SUMMARY AND CONCLUSIONS

5.1 Summary

The objective of this research has been to extend present reliability analysis techniques to conveniently capture sequence dependencies within the framework of a structural decomposition approach. The requirement has been to do this in a way that is accurate while being both less labor intensive and less computationally expensive than combinatorial approaches and component-level Markov models. The motivation for this research is a need for less difficult "first cut" analyses of fault-tolerant systems comprised of fault-tolerant "building blocks."

The methodology is based on defining unique subsystem-level events and computing associated event rates for a given system definition. The event rates are determined from subsystem models and are imbedded in a hierarchical Markov model of the system. The method is derived as it is applied to a specific system definition comprised of two fault-tolerant subsystems.

In formulating a hierarchical model we have made the assumption that the Markov property applies even though we recognize that a hierarchical model is truly semi-Markov as indicated by the unreliability predictions

for the case of dissimilar subsystems (see Figures 4.4 and 4.5). Encouragingly however, the exercises performed indicate that for the given system definition the unreliability prediction of the hierarchical model closely approximates the unreliability prediction of an exact component-level Markov model. The predictions for different subsystems substantiate the claim of semi-Markov behavior. Thus the results of this study show that the hierarchical approach is a viable and useful method for fault-tolerant system reliability modeling. With some additional investigation the applicability of the approach can potentially be broadened. Some additional research topics are discussed below.

5.2 Topics for Further Research

As stated previously, the issue of repair at the subsystem level should be examined in order to extend the usefulness of the hierarchical approach.

The effects of the hierarchical model's semi-Markov behavior can be studied further if three subsystems are considered in the system definition. This would produce a hierarchical model which would have a cascade of local time dependencies for the holding time distributions of a hierarchical state.

When architectures comprised of different subsystems are addressed, error is introduced due to the approximations inherent in the hierarchical approach. Although the error in our exercises was acceptably small, the approximations were not consistently conservative or optimistic. Thus, two problems should be examined. First, given the present hierarchical approach, a measure of the error which does not require a exact model should be produced. Second, a methodology to force the error to be either conservative or optimistic should be investigated. The associated results would consequently bound the system unreliability.

The hierarchical modelling technique used in this study should be further examined to investigate the impact on state probability predictions for subsequent performability analyses.

5.3 Conclusion

Although the hierarchical approach has not been investigated completely, the concept has proved to be viable and to have several useful features with respect to the analysis of large complex systems comprised of fault-tolerant building blocks. Through the use of structural decomposition, the model formulation for such systems is simplified. Furthermore due to the high degree of state aggregation intrinsic to the hierarchical approach, model solution is less computationally burdensome. The computational savings increase, in comparison to a compo-

nent-level modelling approach, as the number and complexity of the subsystems increases. Additional savings occure when some or all of the subsystems are equivalent.

APPENDIX A. TIME-VARYING COMPONENT FAILURE RATES

Throughout this thesis, the component failure rates have been assumed to be constant. This is a large restriction and should be examined. It is desirable to give the component failure rates an arbitrary distribution. The Weibull distribution is commonly used {Ref.(2)}.

The Weibull distribution can model a constant failure rate or monatonically increasing or decreasing failure rates. This probability density function takes the form:

$$f(t) = kmt^{m-1}\exp(-kt^{m}) \qquad t \geq 0 \qquad \text{(A1.1)}$$

$$= 0 \qquad t < 0 \qquad \text{(A1.2)}$$

where k and m are positive, non-zero real numbers. The failure rate corresponding to this distribution is:

$$\lambda(t) = kmt^{m-1} \qquad \text{(A1.3)}$$

From equation A1.3 we see that if m = 1 the Weibull distribution produces a constant failure rate.

A preliminary result indicates that the hierarchical approach models system reliability adequetly, and introduces little error with respect to the truth model (see figure A1.1).

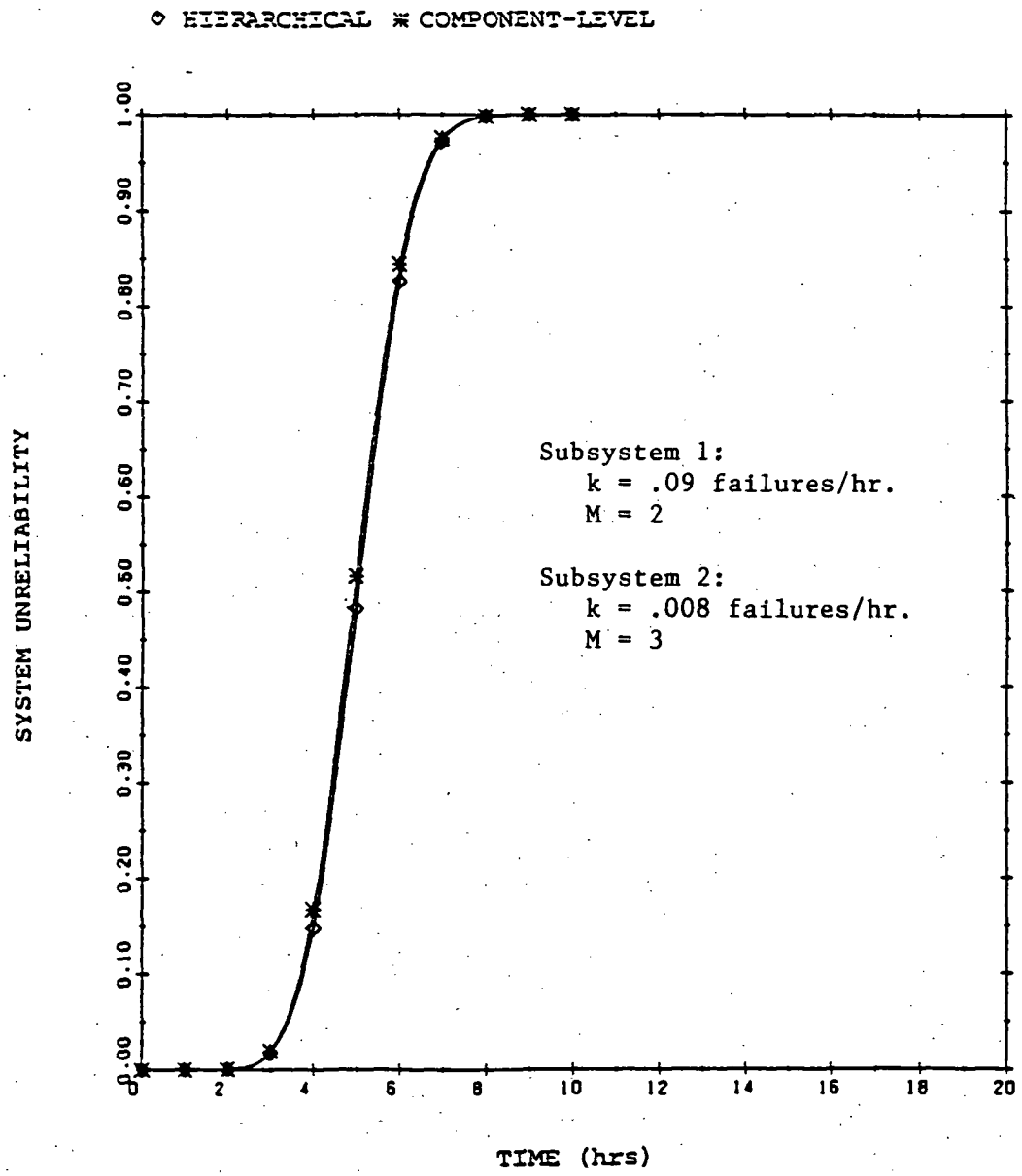


Figure A1.1
Time-Varying Component
Failure Rate Result

REFERENCES

1. Weinstein, W.W., R.S. Schabowsky, and E. Gai, Digital Fly-By-Wire Flight Control System for a Tilt-Rotor Aircraft: Architecture and Reliability Studies, The C. S. Draper Laboratory, Inc., Cambridge, MA, July 1983.

2. VanderVelde, W.E., B.K. Walker, et al., 16.321 Fault Tolerant Control Systems Class Notes, Massachusetts Institute of Technology, Spring Semester, 1984.

3. Shooman, M.L., Probabilistic Reliability: An Engineering Approach, McGraw-Hill, New York, New York, 1968.

4. Siewjorek, D.P., and R.S. Swarz, The Theory and Practice of Reliable System Design, Digital Press, Bedford, MA, 1982.

5. Schabowsky, Jr., R.S., et al., "Evaluation Methodologies for an Advanced Information Processing System," AIAA/IEEE 6th Digital Avionics Conference, Baltimore, MD, December 1984.

6. Luppold, R.H., E. Gai, and B.K. Walker, "Effects of Redundancy Management on Reliability Modelling," Proc. 1984 Automatic Control Conference, June 1983, pp. 1763-1770.

7. Kleinrock, L., Queueing Systems Volume I: Theory, John Wiley & Sons, 1975.

8. Trivedi, K.S., and R.M. Geist, A Tutorial on the Care III Approach to Reliability Modelling, NASA CR-3488, December 1981.

9. Butler, R.W., The Semi-Markov Unreliability Range Evaluator (SURE) Program, NASA Technical Memorandum 86261, July 1984.

10. White, A.L., An Approximation Formula for a Class of Markov Reliability Models, NASA CR-172290, January 1984.

11. Smith. L.O., Introduction to Reliability in Design, McGraw-Hill, New York, New York, 1976.

12. Rice, J.R., Numerical Methods, Software and Analysis, McGraw-Hill, New York, New York, 1983.

13. Lee, L.D., Reliability Bounds for Fault-Tolerant Systems with Competing Responses to Component Failures, NASA TP-2409, 1985.

14. Alger, L.S., and J.H. Lala, "A Real Time Operating System for a Nuclear Power Plant Computer," Seminar: Power Plant Digital Control and Fault-Tolerant Microcomputers, Scottsdale, AZ, April, 1985.