An Extended Stochastic Petri Net (ESPN) model, useful for modeling systems which exhibit concurrent, asynchronous, or nondeterministic behavior is developed. Applications demonstrating the flexibility of the model for a variety of system modeling applications are presented. Analytic techniques for the representation of a class of ESPNs as Markov or semi-Markov processes are discussed, as is the simulation of more general models. Finally, DEEP (the Duke ESPN Evaluation Package) is previewed.
Extended Stochastic Petri Nets:
Applications and Analysis

Joanne Bechta Dugan
Kishor S. Trivedi
Robert M. Geist
Victor F. Nicola

Approved for public release; distribution unlimited.
Extended Stochastic Petri Nets: Applications and Analysis

Joanne Bechta Dugan
Department of Electrical Engineering
Duke University

Kishor S. Trivedi
Department of Computer Science
Duke University

Robert M. Geist
Department of Computer Science
Clemson University

Victor F. Nicola
Department of Computer Science
Duke University

ABSTRACT

An Extended Stochastic Petri Net (ESPN) model, useful for modeling systems which exhibit concurrent, asynchronous, or nondeterministic behavior is developed. Applications demonstrating the flexibility of the model for a variety of system modeling applications are presented. Analytic techniques for the representation of a class of ESPNs as Markov or semi-Markov processes are discussed, as is the simulation of more general models. Finally, DEEP (the Duke ESPN Evaluation Package) is previewed.

1. Introduction

A Petri Net is an abstract, formal graph model useful for representing systems which exhibit concurrent, asynchronous, or nondeterministic behavior. The analysis of the Petri net model provides information about the system it represents, provided the model is a valid representation of the system under study, and the solution of the model is correct. Increasing the flexibility of the modeling tool increases the validity of the model, but makes the model correspondingly more difficult to solve. The goal of this paper is twofold. First, an Extended Stochastic Petri Net (ESPN) model is developed, and the flexibility of such a model is demonstrated through a variety of system modeling applications. Second, analytic and simulation techniques for the solution of an ESPN are derived and demonstrated in DEEP (the Duke ESPN Evaluation Package).

Supported in part by the NASA Langley Research Grant NAG-1-70, by the National Science Foundation grant MCS83-0200, by the Army Research Office grant DAAG29-84-K-0045, and by the Air Force Office of Scientific Research.
Recall the composition of a Petri Net (PN) bipartite graph [Pete81]: a set of places, \( P \) (drawn as circles), a set of transitions, \( T \) (drawn as bars), and a set of directed arcs, \( A \), which connect transitions to places or places to transitions. Places may contain tokens (drawn as dots). The state of a PN, called the PN marking, is defined by the number of tokens contained in each place.

A place is an input to a transition if an arc exists from the place to the transition; a place is an output from a transition if an arc exists from the transition to the place. A transition is enabled when each of its input places contains at least one token. Enabled transitions can fire, by removing one token from each input place and placing one token in each output place. Thus the firing of a transition causes a change of state (produces a different marking) for the PN.

A Stochastic Petri Net (SPN) [Moll82] is obtained by associating with each transition a so-called firing time. Once a transition is enabled, an exponentially distributed amount of time elapses. If the transition is still enabled, it will then fire. A Generalized Stochastic Petri Net (GSPN) [Mars84] allows immediate (zero firing time) as well as timed transitions (exponentially distributed firing times); immediate transitions are drawn as thin bars, timed transitions as thick bars.

An Extended Stochastic Petri Net allows firing times to belong to an arbitrary distribution. In addition to the general firing time distributions, some other extensions to Petri Nets are considered here. An inhibitor arc [Pete81] from a place to a transition has a small circle rather than an arrowhead at the transition. The firing rule is changed as follows: A transition is enabled when tokens are present in all of its (normal) input places, and no tokens are present in the inhibiting input places. When the transition fires, the tokens are removed from the normal input places and deposited in the output places as usual, but the number of tokens in the inhibiting input place remains zero. A probabilistic arc from a transition to a set of output places deposits a token in one (and only one) of the places in the set. The choice of which place receives the token is determined by the probability labels on each branch of the arc. In Figure 1, when the transition is enabled, it fires by removing the token from the input place, and depositing it in either place \( P1 \) (with probability \( a \)) or in place \( P2 \) (with probability \( 1-a \)).

A counter arc from a place to a transition is labeled with an integer value, \( k \). This changes the firing rule such that a transition is enabled when tokens are present in all of its (normal) input places and at least \( k \) tokens are present in the counter input place. When the transition fires, one token is removed from each normal input place, while \( k \) tokens are removed from the counter input place. Associated with a particular counter arc can be a counter-alternate arc, which enables an alternate transition when the count is between 1 and \( (k-1) \), inclusive. The alternate transition can fire each time a token is deposited in the counting input place until there are \( k \) tokens present. The count
remains unchanged by the firing of the alternate transition, as it removes no token from
the counter input place. A counter-alternate arc is labeled with a \( k \).

Neither the counter arc nor the counter-alternate arc are true extensions to Petri
Nets, as both can be realized by a cascade of normal places and transitions [Duga84].
Rather they are useful shorthand notations for such a cascade.

2. ESPN Applications

2.1. Modeling Imperfect Coverage in Fault-Tolerant Computer Systems

The ESPN model was initially developed as an aid in modeling imperfect coverage in
fault-tolerant computer systems [Geis84] for the HARP (the Hybrid Automated Reliability
Predictor) project [Geis83] at Duke University. The HARP reliability model is character-
ized by a behavioral decomposition [Triv83] of the overall model into separate fault-
handling and fault-occurrence submodels. This technique is based on the observation
that the fault-occurrence behavior of a system is composed of relatively infrequent
events while fault-handling behavior is composed of relatively frequent events. Faults
may occur over periods of days or even weeks, but detection and recovery may take only
seconds. The fault-handling model is solved for coverage factors which are then used as
inputs to the overall model. The overall model, which accepts these inputs as well as
parameters defining system structure and fault-occurrence behavior, is then solved for
the system reliability as a function of time.

Fault handling begins when a fault occurs (entry /) and completes when the fault is
handled (exits R or C) or when the system fails (exit F). Exit R represents the correct
recognition and handling of a transient fault (called transient restoration \( t \)), while exit C
represents the reconfiguration of the system to tolerate a permanent or "leaky" transient
fault (traditionally called coverage \( t \)). Here / represents the initial marking of the net.

Many issues must be considered in the design of a general fault-handling model.
Among these are the different classes of faults, the available recovery mechanisms, and
the various possibilities for reconfiguration. The inherent concurrency between the fault
activity and the system fault treatment mechanism can be captured most effectively in
terms of an ESPN. As an example, consider the HARP fault-handling model shown in Fig-
ure 2. The initial marking of this net contains a token in the place labeled 'System OK.'
When a fault occurs in the system, a token is deposited in the place labeled 'Fault.' The
tokens in these two places enable transition T1. The transition fires immediately, thus
removing a token from the input places. A token is then deposited in place 'Active Inter-
mittent' or 'Transient', with probability \( 1-t \) and \( t \) respectively, depending upon whether
the fault is intermittent or transient. (\( t \) is a user-input value defining the percentage of
faults which are transient). Simultaneously, a token is deposited in place 'Lurking', which
represents the presence of a lurking (undetected) fault. If the fault is intermittent, the
token which was deposited in place 'Active Intermittent' will circulate between places
'Active Intermittent' and 'Benign Intermittent,' signifying the oscillation of the fault
between the active and benign states. If the fault is transient, eventually the token which
was deposited in place 'Transient' will be passed to place 'Transient Gone,' signifying the
disappearance of the fault. Note that if a token exists in both places 'Transient Gone'
and 'Lurking,' that transition T5 can fire. This represents a transient fault which disappears
before its presence is felt.

While the fault is lurking and is still active (i.e. a token in place 'Lurking' and no
token in either places 'Benign Intermittent' or 'Transient Gone'), two things may happen:
an error may be produced or the fault may be detected directly. These two events are
represented by transitions T6 and T7, respectively. If the self-test procedure is run while
the fault is active, it will be detected with probability \( d \) (\( d \) is a user-input value defining
the detectability of stuck-at faults). Once an error is produced, it is detected with proba-
bility \( q \), or else it propagates through the system, causing a system failure.

Once the fault is detected, a token is deposited in place 'Counter' which records the
number of times transient recovery is attempted. As long as there are fewer than \( k \)
tokens in place 'Counter,' transient recovery can begin. When recovery is completed, the
fault may still exist, and the detection/recovery cycle may repeat. If recovery has com-
pleted, and the transient fault is gone, T5 is enabled, and the system is once again func-
tioning correctly. If the recovery has completed, and the intermittent fault has gone
benign, transitions T6 and T7 wait for the fault to become active again before they are
enabled.

If the fault is detected too often (more than \( k \) times), the fault is then assumed to be
permanent in nature, and no automatic recovery process is begun. This is modeled by
the accumulation of \( k \) tokens in place C. Once \( k \) tokens are present transition T11 is dis-
abled (transient recovery procedures are inhibited) and transition T12 is enabled (per-
manent recovery procedures begin). Once the fault is determined to be permanent, a
diagnosis procedure is invoked to isolate the faulty unit; this is represented by a token in
place 'Isolate'. The diagnosis procedure is successful with probability \( i \). If the faulted unit
is isolated, the system attempts automatic reconfiguration, which is represented by place
'Reconfigure.' Reconfiguration is successful with probability \( r \) and the token is passed to
place 'OK Degraded', which represents the system again operating correctly, although
performance may be somewhat degraded.

The user of this model must define the distributions for each timed transition, the
probability of fault detection \( (d) \), error detection \( (q) \), isolation \( (i) \) and reconfiguration
\( (r) \). The user must also provide the number of attempts at transient recovery \( (k) \), and
the percentage of faults which are transient \( (t) \).
Let $P_{IC}(\tau)$ denote the probability of depositing a token in place "OK degraded" in an amount of time $\leq \tau$ from the time of entry into the model. Likewise, $P_{IR}(\tau)$ represents re-depositing a token in the place "System OK," and $P_{IF}(\tau)$ represents depositing a token in the "System Failure" place. Let $P_{IR}(\infty)$ (where $\infty \in \{R, C, F\}$) denote the probability of depositing a token in the appropriate exit place,

$$P_{IR}(\infty) = \lim_{\tau \to \infty} P_{IR}(\tau)$$

and let $F_s(\tau)$ be the distribution of times-to-exit,

$$F_s(\tau) = \frac{P_{IR}(\tau)}{P_{IR}(\infty)}$$

The solution of the ESPN model should provide the imperfect distribution $P_{IR}(\tau)$ or the exit probability $P_{IR}(\infty)$ and the time-to-exit distribution $F_s(\tau)$. This set of metrics is then aggregated into the overall model by using either a first-order approximation [Triv84a, Triv84c] or by using exact aggregation [Geis84, Triv84a].

### 2.2. Modeling of Gracefully Degrading Systems

In the dynamic redundancy techniques used in many ultra-reliable systems [Siew82], redundant units are used for error detection, correction, and/or replacement of failed units. They perform no useful work until they replace a failed on-line unit. Graceful degradation techniques, on the other hand, use the redundant hardware as part of the system's normal resources at all times, to increase performance as well as system reliability. The analysis of such a system must deal simultaneously with aspects of performance, fault-tolerance, imperfect coverage, and repair. The solution of such a model would include measures of the "abilities" of the system: reliability, availability and a combination of reliability and performance.

The ESPN representation of a gracefully degrading system with one component type is shown in Figure 3. The number of tokens in place $p_1$, $i$, represents the number of identical units that are operational. The initial number of tokens in place $p_1$, $N$, equals the total number of units. Assuming an exponential failure law (for simplicity of explanation), they fail at rate $i\lambda$ ($\lambda$ is the failure rate of a single unit). Transition $t_1$ represents units failing.

When a unit fails, a single-entry, three-exit fault-handling model (such as the HARP fault-handling model) is entered. The three exits from the fault-handling model, $R$ (transient restoration), $C$ (permanent fault coverage), and $F$ (single point failure), are represented by transitions $t_2$, $t_3$ and $t_4$ respectively. The firing time distributions for these transitions are $P_{IR}(\tau)$, $P_{IC}(\tau)$ and $P_{IF}(\tau)$, respectively, from the solution of the fault-handling model. If we are using a single-fault model (such as the HARP model), we may assume (conservatively) that the occurrence of a second fault during the handling of the first fault causes immediate system failure. This is represented by the counter arc.
enabling transition $t_6$, in which $k_1 = 2$. If one is using a double fault model, in which the third fault causes failure, $k_1$ would then be 3.

Transition $t_2$ returns the token to place $p_1$, and the system continues operating with no loss of performance. Transition $t_2$ represents the reconfiguration of the system to bypass a faulted unit, so a token is deposited in place $p_2$. The failed unit can then undergo some manual repair, and be returned to the active pool of resources. Transition $t_6$ represents the repair of a failed unit while the system is still operational. The repair distribution while the system is up, $F_{RD}(\tau)$, is the firing time distribution for transition $t_6$. If $k_2$ units are down at any given time, the system fails (transition $t_7$).

Once the system has failed, the entire system is taken off-line and repaired. Thus, any tokens that exist in places $p_1, p_2$ or $p_3$ must be moved to place $p_4$ upon system failure. This "flushing out" of places $p_1, p_2$ and $p_3$ is accomplished by immediate transitions $t_9, t_{10}$ and $t_{11}$. The repair distribution while the system is down, $F_{RD}(\tau)$, is the firing time distribution for transition $t_6$. When the system is repaired, all $N$ tokens are deposited in place $p_1$, thus there are $N$ arcs from transition $t_6$ to place $p_1$.

The solution of this model yields measures of the "abilities" of the system.

Reliability, the probability that the system has not failed by time $t$, is given by

$$R(t) = 1 - \text{Prob}[ \text{token reached place } p_4 \text{ by time } t ]$$

Availability, the probability that the system is up at time $t$, is given by

$$A(t) = 1 - \text{Prob}[ \text{token in place } p_4 \text{ at time } t ]$$

The steady-state availability, the long-term probability that the system is up, is given by

$$A_\infty = 1 - \text{Prob}[ \text{token in place } p_4 \text{ in steady-state } ]$$

The expected computation capacity at time $t$ [Triv84b], assuming that each unit has a computation capacity of $\alpha$, is given by

$$E[C_t] = \sum_{i=1}^{N} i \cdot \alpha \cdot \text{Prob}[ i \text{ tokens in place } p_1 \text{ at time } t ]$$

The expected accumulated computation capacity (termed accumulated reward in [Triv84b]) at time $t$, can be obtained by an additional integration:

$$E[Y_t] = \int_{0}^{t} E[C_x] \, dx$$

Thus, an ESPN model of a gracefully degrading system, besides being very easy to understand, is general enough to provide measures of reliability, availability, and performance.
3. ESPN Analysis

3.1. The Reachability Tree

The first step in the analysis of any Petri Net is the generation of the reachability tree. A marking \( M' \) is said to be immediately reachable from \( M \) if the firing of some transition \( T \), which is enabled in \( M \), yields \( M' \). \( M' \) is reachable from \( M \) if it is immediately reachable from \( M \) or is reachable from any marking which is immediately reachable from \( M \) or is \( M \) itself.

The nodes of the reachability tree represent reachable markings of the net; the root node represents the initial marking. A directed edge points from marking \( M \) to marking \( M' \) if \( M' \) is immediately reachable from \( M \). The edge is labeled with the transition \( T \) whose firing produces \( M' \) from \( M \), and the probability \( p \), that \( M' \) is reached from \( M \) when \( T \) fires.

As an example of the generation of a reachability tree, consider the submodel of the HARP fault-handling model shown in Figure 4. The reachability tree for this net is shown in Figure 5. Each marking in the reachability tree is labeled with the names of the places which contain a token in that marking.

A reduction of the reachability tree is possible, by partitioning markings into two classes, and absorbing markings of one class into the other. A marking is called a vanishing marking [Mars84] if it enables an immediate transition. A vanishing marking is so named since no time is spent in this marking. If a marking enables only timed transitions then it is called a tangible marking. A vanishing marking can be absorbed into the tangible marking that precedes it, by adjusting the next-state and probability labels on the edges. Figure 6 represents the reduced reachability tree of Figure 5. It is on this reduced tree that the analysis is performed.

3.2. Markovian Reachability Tree

DEFINITION (Markovian Reachability Tree)

A reduced reachability tree can be called Markovian if it exhibits the Markov property, that is, if all firing time distributions for timed transitions are exponential.

THEOREM 1: A Markovian reachability tree can be classified as a Markov chain, in which each state in the Markov chain represents a unique marking in the reachability tree.

Proof: See Molloy [Moll81], Natkin [Natk80], and Marsan, Balbo, and Conte [Mars84] who have developed this theory as (Generalized) Stochastic Petri Nets. Figure 7 presents an illustration of the relationship between an ESPN whose reachability tree is Markovian, and the resulting Markov chain. In the ESPN, each timed transition is labeled with its firing rate. In the Markov chain, the initial state is state \( TL \) with probability \( t \), and \( AL \) with pro-
3.3. Semi-Markovian Reachability Tree

DEFINITION (Semi-Markovian Reachability Tree)

A reduced reachability tree can be called semi-Markovian if it exhibits the Markov property at the times when marking changes occur.

Three conditions concerning the transitions in the ESPN must be satisfied for the reachability tree to be semi-Markovian. Before we study these conditions, we need to classify each of the timed transitions into one of three groups: exclusive, competitive, or concurrent.

Exclusive Transition -- A timed transition \( T_i \) is said to be exclusive if, for every marking \( M_k \) in the reduced reachability tree that enables \( T_i \), \( M_k \) enables no other transition. That is, whenever transition \( T_i \) is enabled, no other transition is enabled.

Competitive Transition -- Let \( T_i \) be a non-exclusive timed transition. Then there exists a marking \( M_k \) in which \( T_i \) and some other transition \( T_j \) are enabled. If for every such \( T_j \) in every such marking \( M_k \), the firing of \( T_j \) disables the transition \( T_i \), then \( T_i \) is called a competitive transition.

Concurrent Transition -- Again let \( T_i \) be a non-exclusive timed transition. Then there exists a marking \( M_k \) in which \( T_i \) and some other transition \( T_j \) are enabled. If for any such \( T_j \) in any such marking \( M_k \), the firing of \( T_j \) does not disable transition \( T_i \), then \( T_i \) is called a concurrent transition.

THEOREM 2: A reachability tree is called semi-Markovian if it satisfies three conditions:

Condition 1: The firing time of an exclusive transition may belong to any arbitrary probability distribution.

Condition 2: The firing time of a competitive transition may belong to any arbitrary probability distribution. However, the firing time of a transition that is re-enabled subsequent to being disabled is assumed to be of the type preemptive-repeat-different. That is, the time between the enabling and firing of the re-enabled transition is independent of and has the identical distribution as the preempted firing time.

Condition 3: The firing time of all concurrent transitions must be exponentially distributed.

Proof: It is necessary to verify that a semi-Markovian reachability tree satisfies the Markov property at the times at which state changes occur. Recall that a state represents a marking for the ESPN, and that state changes occur when transitions fire. In examining the markings for the reachability tree, it is useful to distinguish three cases.

Case 1: The marking enables an exclusive transition.
The time spent in the marking is the time needed for the exclusive transition to
fire and is independent of the past history of the process.

Case 2: The marking enables non-exclusive transitions.

Assuming (without loss of generality) that the marking enables both a compe-
titive and a concurrent transition, the future of the process depends on which
fires first. If the competitive transition fires first, then the concurrent transi-
tion may still be enabled upon entry into the next state. In this next state, the
remaining time for the concurrent transition depends on the time needed for
the competitive transition to fire in the previous state. The memoryless prop-
erty of the exponential distribution assures us that this remaining time distrib-
ution will be identical to the original firing time distribution.

If the concurrent transition fires first, then by definition, the competitive transi-
tion is disabled. If the process subsequently enters another marking in which
the competitive transition is re-enabled, the preemptive-repeat-different
assumption of condition 2 assures us that the firing time of a re-enabled transi-
tion is identical to the original firing time distribution, and is independent of the
preemption.

Case 3: The marking enables no transition.

If a marking enables no transition, then this marking is an absorbing state of
the process, and no further state changes may occur.

Theorem 3: A semi-Markovian reachability tree can be classified as a semi-Markov pro-
Pess [Fell64], in which each state in the semi-Markov process represents a unique mark-
ing in the reachability tree.

Proof: In the ESPN, the firing time distribution $T_K(\tau)$ is the probability that transition $K$
fires in an amount of time $\leq \tau$ after it is enabled. Let $t_k(\tau)$ be the corresponding density
function. In any subsequent diagrams, a transition will be labeled with its corresponding
distribution.

In the semi-Markov process, the defective probability distribution $F_i(\tau)$ (with
$F_i(0)=0; F_i(\infty)=1$), is the probability that a sojourn time in state $i$ has duration $\leq \tau$ and
ends by a jump to state $j$. The next-state transition probability $a_{ij} = F_j(\infty)$.

The unconditional sojourn distribution in state $i$ is the sum of the conditional
sojourn time distributions:

$$S_i(\tau) = \sum_j F_{ij}(\tau).$$

We begin our analysis at some marking $M_i$ (called state $i$ in the semi-Markov pro-
cess). Suppose the firing of transition $T_j$ from marking $M_i$ yields marking $M_j$, where $T_j$ is
an exclusive transition. Then the conditional sojourn time distribution for this state is simply the transition firing time distribution.

\[ F_{ij}(\tau) = T_j(\tau) \]

Next suppose that, from marking \( M_i \), the firing of transition \( T_j \) yields \( M_j \) with probability \( p_j \), and the firing of transition \( T_k \) yields \( M_k \) with probability \( p_k \), and that no other markings are immediately reachable from \( M_i \). (Note that if \( T_j \neq T_k \) then the two transitions are competing and \( p_j = p_k = 1 \). If \( T_j = T_k \) then after the firing of the transition, a probabilistic branching occurs, where \( p_j + p_k = 1 \).) If \( T_j \neq T_k \) then,

\[ F_{ij}(\tau) = \int_0^\tau (1-T_k(x)) \cdot t_j(x) \, dx \quad \text{and} \quad F_{ik}(\tau) = \int_0^\tau (1-T_j(x)) \cdot t_k(x) \, dx. \]

If \( T_j = T_k = T \) then,

\[ F_{ij}(\tau) = p_j \cdot T(\tau) \quad \text{and} \quad F_{ik}(\tau) = p_k \cdot T(\tau). \]

The conditional sojourn time calculation generalizes to markings that enable more than two transitions. Let \( A \) be the set of enabled transition in marking \( M_i \). Let \( a_{ij} \in A \) be the transition whose firing causes a jump from state \( i \) to state \( j \) with probability \( p_{ij} \). Then the conditional sojourn time distribution is:

\[ F_{ij}(\tau) = p_{aij} \cdot \int_0^\tau \prod_{e \in a_i \cup a_{ij}} (1 - T_e(x)) \cdot t_{aij}(x) \, dx. \]

4. Conversion of an Acyclic Reachability Tree to a Semi-Markov Process

If the firing time of a concurrent transition is not exponentially distributed, then the corresponding ESPN cannot be converted to a semi-Markov process using Theorem 3. But it may be possible to convert the reduced reachability tree to a semi-Markov process by a judicious lumping of markings to form one state. The conditional sojourn time distributions can then be determined by performing a path analysis of the markings in the lumped state.

**DEFINITION (Acyclic reachability tree)**

A reduced reachability tree can be termed *acyclic* if each marking can be visited only once; that is, if there are no cycles. Formally, for every marking \( M' \) that is reachable from a marking \( M, M \) must not be reachable from \( M' \).

**DEFINITION (Concurrency set)**

For each concurrent transition \( T \) whose firing time is generally distributed, (i.e. not exponential), define a *concurrency set*, \( C_T \), such that a marking \( M \) is an element of the concurrency set \( C_T \) if any of the following conditions are satisfied:

1) \( M \) enables transition \( T \),
2) a marking \( M_j \) (\( \in C_T \)) is reachable from some \( M_k \) (\( \in C_T \)) through \( M \),
3) \( M \) is the "closest" marking such that each \( M_j \) (\( \in C_T \)) is reachable from \( M \). ("Closest" in the sense that there is no \( M' \) such that \( M' \) is reachable from \( M \) and each \( M_j \) (\( \in C_T \)) is reachable from \( M' \).)
4) \( M \) is an element of some concurrency set whose intersection with \( C_T \) is nonempty.
Once the concurrency sets have been determined, the conversion of the reachability tree to a semi-Markov process may proceed. A marking \( M \) that is an element of no concurrency set becomes a state in the semi-Markov process, and the calculation of the sojourn times proceeds as in the semi-Markovian reachability tree.

For each concurrency set, a state in the semi-Markov process is formed by combining all markings in the set into a single state. The determination of the conditional sojourn time distributions in this state consists of calculating, for each possible path through the lumped state, the time-to-exit for each output arc. This path analysis is best explained through a series of simple examples.

Consider the ESPN in Figure 8a, where \( T_A \) and \( T_C \) have general distributions, while \( T_B \) is exponential. The corresponding reachability tree is shown in Figure 8b. In this example, \( T_A \) is an exclusive transition, while \( T_B \) and \( T_C \) are concurrent transitions. The concurrency set associated with transition \( T_C \) contains markings \( BC \) and \( DC \), the state containing these markings will be labeled \( C \). The semi-Markov process representation of this tree is shown in Figure 8c. The conditional sojourn time distributions for the merged state \( C \) are given by:

\[
F_{C,BF}(\tau) = \int_0^\tau (1 - T_B(x)) t_C(x) \, dx
\]

\[
F_{C,DE}(\tau) = \int_0^\tau T_B(x) t_C(x) \, dx
\]

Clearly, as the concurrency increases, the complexity of the path analysis also increases. The level of complexity is increased further when we consider a sequence of concurrent transitions, some of which are not enabled immediately upon entering the merged state. As an example, consider the ESPN in Figure 9a, where concurrent transition \( T_B \) is generally distributed. The corresponding reachability tree and semi-Markov process are in Figures 9b and 9c, respectively. Since markings \( BC, BE, BF \) and \( BG \) each enable transition \( T_B \), they will be merged into a single state, called \( B \). There are four possible exits for this state, each corresponding to a distinct path.

**Path 1: BC→DC.** The probability that state \( DC \) is entered at time \( x \) is simply the probability that \( T_C \) has not fired by time \( x \) and that \( T_B \) fires at time \( x \).

\[
F_{B,DC}(\tau) = \int_0^\tau (1 - T_C(x)) t_B(x) \, dx
\]

**Path 2: BC→BE→DE.** The probability that state \( DE \) is entered at time \( x \) is the probability that, at some time \( u \), \( T_C \) fires, thus enabling transition \( T_B \). Between \( u \) and \( x \), \( T_B \) does not fire, and at time \( x \), \( T_B \) fires. See Figure 10a for a timing diagram of this sequence.

\[
F_{B,DE}(\tau) = \int_0^\tau \int_0^\tau t_B(x) (1 - T_B(x - u)) t_C(u) \, du \, dx
\]
Path 3: BC→BE→BF→DF. In the timing diagram for this path, shown in Figure 10b, T_C fires at some time w, then T_B fires at some time u≥w, where T_B was enabled at time w. Between u and x T_B does not fire, where T_B was enabled at time u, and at time x, T_B fires.

\[ F_{B,DF}(\tau) = \int_0^x \int_0^u t_B(z) (1 - T_B(z-u)) t_D(u-w) t_C(w) \, dw \, du \, dz \]

Path 4: BC→BE→BF→BG→DG. Figure 10c shows the timing diagram for this path, in which T_C fires at time v, and enables T_B which fires at time w, thus enabling T_P. At some time u, T_B fires. Then at time x≥u≥w≥v, T_B fires, where T_B was enabled at time 0. Thus,

\[ F_{B,DG}(\tau) = \int_0^x \int_0^u \int_0^w t_B(z) t_P(u-w) t_B(w-v) t_C(v) \, dw \, du \, dx \]

This methodology can be validated [Duga84] by looking at the case in which the firing times are all exponential. This system then reduces to a Coxian stage-type distribution.

If cycles are permitted within a lumped state, then an infinite number of possible paths arise, and an automatic conversion of an arbitrary ESPN to a semi-Markov process becomes intractable. Even in acyclic reachability trees it seems infeasible to perform automatic path analysis and solve for the sojourn times on any but the most simple systems. In such cases we can easily resort to simulation of the ESPN to obtain the desired solution.

5. DEEP (The Duke ESPN Evaluation Package)

The design of the Duke ESPN Evaluation Package, DEEP, can be divided into three levels: input, analysis, and solution. (See Figure 11.) DIVE (The DEEP Interactive Video Editor) allows for the graphic input of an ESPN, by allowing the user to position the places, transitions and arcs on the screen. It interprets the net as it is input and checks to see if it is a valid ESPN. Once the net is input, its description is fed to another module, Reach. We are also in the process of developing a textual input language for DEEP.

Reach generates the reachability tree for the net, and absorbs the vanishing markings into the tangible ones. Once the reduced reachability tree is generated, it is characterized as Markovian, semi-Markovian, or neither. If the reduced reachability tree is Markovian, it is solved as a continuous-time Markov chain; if it is semi-Markovian, it is solved as a semi-Markov process. If it is neither, it is simulated.

ESPN-sim uses one of two types of simulation, transient or ergodic, depending on the measures desired by the user. If the user is interested in exit-probabilities and time-to-exit distributions (as in modeling imperfect coverage), or time-dependent occupation probabilities for places (as in reliability modeling) a transient simulation is performed. If the user is interested in long-term or average measures, such as average token count or
transient utilization, an ergodic simulation is performed.

DEEP is undergoing development and testing, and only portions of it have been fully implemented thus far.

6. An Example

As an example of the hierarchical modeling of a gracefully degrading system, we will solve an "instantaneous coverage" [Triv84c] version of the "ability" model discussed in Section 2.2. In this model, assuming that the time spent in the fault-handling model is negligible as compared with fault-occurrence and repair times, transitions $t_2, t_3,$ and $t_4$ are combined into one immediate transition $t_f$ (See Figure 12). The probabilistic output arc labels are functions of $r$ (transient restoration) and $c$ (coverage) from the solution of the fault-handling model. Transition $t_5$ can be eliminated, since we are ignoring the possibility of near-coincident faults. (Methods of incorporating near-coincident faults can be found in [Triv84c] and [Geis84].)

Before we can solve the "ability" model, the fault-handling model (Figure 2) must be solved for the parameters listed in Table 1. The imperfect probability distributions, $P_{IR}$ and $P_{IC}$ are shown in Figure 13. For the solution of the "ability" model, we need only $c = P_{IC}(\infty)$ and $r = P_{IR}(\infty)$.

Considering the system failure state as an absorbing state (i.e. $F_{RD}=0$), the model was simulated for the parameters listed in Table 2. The occupation probabilities for places $p_1$ and $p_4$ are shown in Figure 14. A plot of the reliability of the system is shown in Figure 15a, while Figure 15b shows a plot of the expected computation capacity of the system, assuming that $\alpha=1$.

To estimate the availability of the system, the model was again simulated for the values listed in Table 2. Additionally, off-line repair was allowed, where $F_{RD}(\tau)$ was assumed Normally distributed (truncated at zero; mean = 10 hours, standard deviation = 2 hours). A plot of the estimated availability of the system is shown in Figure 16.

7. Conclusions

The ESPN model greatly enhances the modeling power of stochastic Petri Nets, but also increases the complexity of the solution of the model. We have developed both analytic and simulative solution techniques for ESPNs; the choice of solution technique (which can be made automatic) depends on the characteristics of the net. DEEP (The Duke ESPN Evaluation Package) will provide automated analysis of an arbitrary ESPN, and will be ready for initial testing in late 1984.
8. Acknowledgements

We would like to thank Professor Vidyadhar Kulkarni for many helpful discussions and John White for his work on DIVE. A special thanks is extended to Dr. Mark Smotherman for his collaboration on HARP.

9. References


Figure 1. Petri Net Extensions

inhibitor arc

probabilistic arc

counter arc and counter-alternate arc

Figure 2. ESPN Fault-Handling Model

Figure 3. "Ability" Model of a Gracefully Degrading System

P1: Operational Units
P2: Fault Handling
P3: Failed Units
P4: System Failed
Figure 4. HARP Submodel

Figure 5. Reachability Tree

Figure 6. Reduced Reachability Tree
Figure 7. ESPN and Corresponding Markov Chain

Figure 8.
Figure 12. Instantaneous coverage ability model.
<table>
<thead>
<tr>
<th>Time</th>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVE Transition</td>
<td>unif(0, 1 sec.)</td>
</tr>
<tr>
<td>BENIGN Transition</td>
<td>unif(0, 0.5 sec.)</td>
</tr>
<tr>
<td>Transient Lifetime</td>
<td>exp(100/sec)</td>
</tr>
<tr>
<td>DETECT Transition</td>
<td>unif(0, 0.4 sec.)</td>
</tr>
<tr>
<td>ERROR Transition</td>
<td>weibull(10/sec., 2.5)</td>
</tr>
<tr>
<td>ERROR-DETECT Transition</td>
<td>weibull(50/sec., 0.25)</td>
</tr>
<tr>
<td>ISOLATE Transition</td>
<td>truncated normal(0.0, 1.0)</td>
</tr>
<tr>
<td>RECOVERY Transition</td>
<td>2-stage erlang(100/sec.)</td>
</tr>
<tr>
<td>RECONFIGURE Transition</td>
<td>truncated normal(1.0, 0.5)</td>
</tr>
</tbody>
</table>

**Other Parameters**

- Probability of fault detection by self test: 0.8
- Probability of error detection: 0.45
- Probability of isolating detected fault: 0.5
- Number of recovery attempts: 5
- Probability of successful reconfiguration: 0.75
- Fraction of faults which are transient: 0.5
- Desired confidence level: 90%

**Table 1. Input Parameters for Fault-Handling Model**

---

**Figure 13. Results of Simulation of Fault-Handling Model**

\[ c = P_{IC}(\infty) = 0.1406 \]

\[ r = P_{IR}(\infty) = 0.4992 \]
\[ N = 3 \text{ units} \]
\[ k_2 = 2 \text{ units} \]
\[ \lambda = 10^{-2} \text{ failures/hour} \]
\[ F_{RU} \text{ - 2-stage Erlang (0.5/hour)} \]

**Table 2. "Ability" Model Parameters**

**Figure 14. "Ability" Model Solution**
Figure 15. a) Reliability  b) Expected Computation Capacity