Markov Reward Approach to Performability and Reliability Analysis

K.S. Trivedi* M. Malhotra R.M. Fricks
Dept. of Electrical Eng. AT&T Bell Laboratories Dept. of Electrical Eng.
Duke University Holmdel, NJ 07733 Duke University
Durham, NC 27708-0291

Abstract

Performability and reliability modeling techniques and tools have been an area of intensive research activity in the last ten years. We present a unified mathematical framework for performability and reliability models in terms of Markov reward models. The framework to be presented is not the only one available for performability analysis but it is the most commonly used one. To complete our exposition, we describe two modeling software packages SHARPE and SPNP.

1 Introduction

Fault-tolerant systems provide continuity of service despite component failures. However, the performance delivered by the system may degrade. A very simple example is that of a mirrored disk system. The system operates as long as at least one disk is operational. If both the disks are operational, then the reads are serviced from the disk with minimum seek time. Thus the system delivers higher read performance when both the disks are functioning than when one of the disks has failed. Performability analysis aims to capture this interaction between the failure-repair behavior and the performance delivered by the system.

Significant advances have been made in performability modeling and analysis since Beaudry [3] defined combined measures of performance and reliability and Meyer[18] proposed a general framework for performability analysis. Several new algorithms have been proposed for calculating performability measures and a number of tools have been designed for performability analysis. In this paper, we present a unified framework for performability and reliability analysis using Markov reward models (MRMs). For a more comprehensive account of MRMs for performability analysis we suggest [27]. We, also, present two software packages that allow the specification and solution of MRMs. We conclude the paper by discussing some of the computational problems that arise in performability analysis.

*This work was supported in part by the National Science Foundation under Grant CCR-9108114 and by the Naval Surface Warfare Center under grant N60921-92-C-0161.

2 Markov Reward Models

Let \( \Theta(t), t > 0 \) be a continuous-time finite-state homogeneous Markov chain (CTMC) with state space \( \Psi \). A constant reward rate \( \tau_i \) is associated with each state \( i \) of the Markov chain. With the reward rate specifications, the CTMC can be termed as Markov reward model. If the MRM spends \( \tau_i \) time units in state \( i \), then \( \tau_i \) is the reward accumulated. It is also possible to associate reward rates with the transitions of the CTMC.

Let \( Q \) be the generator matrix and \( P(t) \) be the state probability vector of the MRM. Here \( P(t) \) denotes the probability of the MRM being in state \( i \) at time \( t \). The transient behavior of this MRM is given by the Kolmogorov differential equation:

\[
\frac{dP(t)}{dt} = P(t)Q ,
\]

given the initial state probability vector \( P(0) \). The steady-state probability vector \( \pi \) assuming that it exists and is unique is given by:

\[
\pi Q = 0 ,
\]

subject to the condition \( \sum_{i \in \Psi} \pi_i = 1 \). Here \( \pi_i \) is the steady-state probability of the MRM being in state \( i \). Define a cumulative state probability vector of the MRM as \( L(t) = \int_{0}^{t} P(x)dx \). \( L(t) \) denotes the expected total time spent by the MRM in state \( i \) during the interval \( [0, t] \). From (1), we get:

\[
\frac{dL(t)}{dt} = L(t)Q + P(0) .
\]

For MRMs with absorbing states, the state space \( \Psi \) can be partitioned into two: \( \Psi_A \) (absorbing states) and \( \Psi_T \) (transient states). Corresponding to the transient states, the submatrix \( Q_T \) of \( Q \) can be defined. The mean time spent by the MRM in state \( i \) is given by:

\[
\tau_i = \int_{0}^{\infty} P_i(x)dx ,
\]

which can be computed by integrating (1) from 0 to \( \infty \):

\[
\tau Q_T + P_T(0) = 0 .
\]
The mean time to absorption in such a Markov chain is given by:

$$MTTA = \sum_{i \in \mathcal{F}_T} \tau_i .$$  \hspace{1cm} (5)

For further details on computational approaches see [21].

3 Performability and Reliability Measures

Performability analysis can be naturally adapted to MRMs by suitable assignment of reward rates. Let $\tau(t) = r_\Phi(t)$ be the instantaneous reward rate of the MRM. The accumulated reward over a period of time $[0,t]$ is given by:

$$\Phi(t) = \int_0^t \tau(x)dx = \int_0^t r_\Phi(x)dx .$$  \hspace{1cm} (6)

The expected instantaneous reward rate at time $t$ is:

$$E[\tau(t)] = \sum_{i \in \mathcal{F}} r_i P_i(t) .$$  \hspace{1cm} (7)

The expected reward rate in steady-state is:

$$E[\Phi_s] = \sum_{i \in \mathcal{F}} r_i \tau_i .$$  \hspace{1cm} (8)

The expected accumulated reward in the interval $[0,t]$ is:

$$E[\Phi(t)] = \sum_{i \in \mathcal{F}} r_i L_i(t) .$$  \hspace{1cm} (9)

The expected time-averaged reward in the interval $[0,t]$ is given by $\sum_{i \in \mathcal{F}} P_i(t)/t$. For an MRM with absorbing states, expected accumulated reward until absorption is:

$$E[\Phi(\infty)] = \sum_{i \in \mathcal{F}_T} r_i \tau_i .$$  \hspace{1cm} (10)

The distribution of $T(t)$ is computed as:

$$P[T(t) \leq \psi] = \sum_{r_1 \leq \psi, i \in \mathcal{F}} P_i(t) .$$  \hspace{1cm} (11)

The distribution of accumulated reward until absorption and distribution of accumulated reward over a finite period of time can also be computed.

Let the time to accumulate a given reward $r$ be denoted by $\Gamma(r)$. Then the distribution of $\Gamma(r)$ is known once the distribution of accumulated reward is known:

$$P[\Gamma(r) \leq t] = 1 - P[\Phi(t) < r] .$$  \hspace{1cm} (12)

For example, the distribution of time to complete a job that requires $r$ units of processing time on a system which is modeled by an MRM can be computed in this manner [12]. For details on computational techniques see [27].

In a reliability model, a reward rate of 1 is assigned to all the system operational states and reward rate 0 is assigned to all the system failure states. The instantaneous availability of the system is then $E[T(t)]$ and steady-state availability is $E[\Phi_s]$. The cumulative operational time of the system in time interval $[0,t]$ is $E[\Phi(t)]$. Interval availability is the proportion of time a system is operational in a given interval of time and it is given by $E[\Phi(t)]/t$. Measures related to time to first system failure are also of interest. To compute these measures, all the failure states are made absorbing (outgoing arcs from these states are removed). Reliability is then given by $E[T(t)]$. The lifetime of the system in interval $[0,t]$ is $E[\Phi(t)]$ and mean time to system failure (MTTF) is $E[\Phi(\infty)]$. The repairability of the system is computed by making all the operational states absorbing, reversing the reward rates (i.e., making reward rate 1 to 0 and vice-versa), and computing $E[T(t)]$.

4 The MRM Approach

MRM framework naturally leads to a separation of the performance and reliability models of the system. Performance model is solved to obtain reward rates which are assigned to the reliability model (also known as structural model [17]). However, this separation introduces an approximation in that reward rates assigned to any state of the reliability model are typically constant values (i.e. time-independent). Thus, it is assumed that for the sojourn in a state of the reliability model, the system provides a steady-state performance. The accuracy of this approximation depends upon the differences in the rates at which the events in the performance and the dependability model occur. Smaller the difference, the less accurate the results are. It is for this reason that this approach is said to be based on a time-scale decomposition. In [27], we gave an example where this approach yields inaccurate results.

The MRM approach is not directly applicable if parameters of the reliability model depend on parameters of the performance model (e.g., workload dependent failure rate of a server) and vice-versa. MRM framework is not the only one available for performability analysis but it is the most commonly used one.

5 Tools for Performability and Reliability Analysis Using MRMs

In this section, we focus on two software packages that allow the specification and solution of MRMs for performability and reliability analysis.

5.1 SHARPE

SHARPE (Symbolic Hierarchical Automated Reliability/Performance Evaluator) was originally developed in 1986 by Sahnier and Trivedi [24]. It has been
significantly updated in 1991 [23]. It is written in C and runs on UNIX and VMS. The main feature of SHARPE is that hybrid and hierarchical models can be easily constructed. The overall system model may consist of several submodels of possibly different types. The model types allowed are fault-trees, reliability block diagrams, reliability graphs, Markov chains, acyclic semi-Markov chains, single and multi- chain product forming the networks, GSPNs, and series-parallel task graphs. For example, the reliability of a system can be modeled by a reliability block diagram where reliability of each block is computed by solving a Markov chain. If a single Markov model for such a system was constructed, then it could have a state space of exponential size. Hierarchical modeling alleviates the problems of model largeness and stiffness to a large extent.

The model can be input either in interactive mode or by means of batch files. Submodels may be specified in the syntax of appropriate model types. These submodels may then be used in defining higher level models. Hierarchical is established by passing parameters from lower level models to higher level models as specified by the user. SHARPE allows exponential polynomial \( \sum_{i=1}^{n} a_i x_i e^{\lambda_i} \) to be attached to components (tasks, states). Thus failure distribution associated with basic components in case of combinatorial models (fault-trees, block diagrams, and reliability graphs) or sojourn-time distribution associated with state transitions in a semi-Markov chain could be an exponential polynomial.

The combinatorial models (fault-trees, block diagrams, and reliability graphs) are used for reliability and availability analysis. With each component (or link) a failure-time distribution may be specified as an exponential polynomial (symbolic in \( t \)). SHARPE computes distribution function (symbolic in \( t \)), mean, and variance of time to failure of the system. User may opt for numeric solution in which case a numeric value of the distribution function at a given time \( t \) is computed. If failure probability or availability (instantaneous or steady-state) is specified for each component, then system failure probability or system availability is output. Specialized solution algorithms are used for different model types.

(semi)-Markov chains may be used for both reliability and performance analysis. Only acyclic or irreducible semi-Markov chains are allowed. The sojourn-time distribution in case of a semi-Markov chain could be exponential polynomial. Reward rates may be associated with states of a (semi)-Markov chain and reward based performance measures can be computed. These reward rates may be specified either as a numeric value or as the output of a performance submodel specified by a queuing network, series-parallel task graph, (semi)-Markov chain, or a GSPN. In this case, the performance model is first solved to obtain a numeric value for the reward rate. For irreducible chains, SHARPE computes expected steady-state reward rate. For chains with an absorbing state, SHARPE computes distribution of accumulated reward until absorption, expected instantaneous reward rate, and expected accumulated reward until a given time \( t \).

For irreducible Markov chains, SOR or Gauss-Seidel is used for steady-state solution. For transient analysis, uniformization with steady-state detection is used. Symbolic solution of state probabilities is obtained using Laplace transform and partial fraction expansion approach. For fault-trees and reliability graphs, a disjoint sum of products algorithm is used.

5.2 SPNP

SPNP (Stochastic Petri Net Package) has been developed by Ciardo et al [7]. It is written in C and runs on a variety of operating systems including UNIX, AIX, OS/2, and VMS. The model type used for input is a stochastic reward net (SRN). SRNs incorporate several structural extensions to GSPNs [2] such as marking dependencies (marking dependent arc cardinalities, enabling functions, etc.) and allow reward rates to be associated with each marking. The reward function can be marking dependent as well. SRNs are specified using CSPL (C based SPN Language) which is an extension of C with additional constructs for describing the SPN models. Whereas CSPL exploits the full power of C and makes the SRN specification very flexible, it also makes it imperative that the user knows C. There is no interactive interface, but a graphical interface exists. The user can either specify the SRN graphically or type in the CSPL file.

SRN specifications are automatically converted into an MRM which is then solved to compute a variety of transient, steady-state, cumulative, and sensitivity measures. Standard measures such as average number of tokens in a place, average throughput of a timed transition, probability (transient or steady-state) that a place is not empty, and probability that a timed transition is enabled can be computed. These basic measures are combined or interpreted (according to the user) to yield various reliability, performance, and performance measures. For example, suppose that a token in place \( p \) implies that the system has failed. Then the probability of a token in place \( p \) at time \( t \) is the unreliability of the system. For a given reward function, expected value of the function, expected accumulated reward over a finite time interval, and time-averaged expected value can be computed. For SRNs with absorbing markings, mean time to absorption and expected accumulated reward until absorption can be computed.

For transient analysis, a highly accurate and efficient version of uniformization is employed. This incorporates steady-state detection of the underlying discrete-time Markov chain and computation of Poisson probabilities using Fox and Glynn's method. For steady-state solution, Gauss-Seidel or near-optimal SOR method is used. Sensitivity analysis may also be carried out by computing the derivatives of some of the output measures w.r.t. model parameters. Hierarchical models can be constructed by parameter passing from lower level models (subnets) to higher level models (subnets). Iterative solution approach based
on fixed-point iteration can be employed by parameter passing via a shell file. State truncation is also easy to incorporate.

6 Computational Problems, Extensions, and Current Trends

In actual practice, performability and reliability analysis is plagued with several computational problems. We discuss some of these.

6.1 Largeness

Largeness of state-space is a major problem for Markovian models. To overcome this, we may resort to either largeness-tolerance or largeness-avoidance schemes. In the former approach, a large overall model is automatically generated starting with a more concise description [9]. The model is solved using sparse storage techniques and sparsity preserving efficient numerical methods. In largeness-avoidance approach, the model size is reduced during model generation and therefore a large model is not generated. State truncation methods [19], hierarchical model solution [16], and hybrid models that judiciously combine different model types [11] are examples of largeness avoidance.

6.2 Stiffness

Stiffness of the model is another problem that plagues its solution. It arises mainly due to large difference in failure and repair rates or failure and job arrival rates. The MRM approach reduces stiffness by separating the performance and reliability models but the stiffness within reliability model remains. Stiffness may be avoided by using aggregation [6] that yields approximate solution. To tolerate stiffness, special stable stiff solvers may be used [14, 22].

6.3 Non-Markovian Models

For analyzing non-Markovian models, there are three different approaches:

- Discrete-event simulation of the non-Markovian model. The reader can easily obtain the vast literature on this subject.

- Phase-type expansion of the non-exponential distributions. There have been several efforts to analyze non-Markovian reliability models using phase type expansion [25]. For other references, see [10, 15, 26]. For performability analysis of non-Markovian models, there have been a few efforts in the past. Bobbio [4] has proposed how SPNs with generally distributed firing times can be used to generate stochastic reward models for performability analysis. Cumani [8] has implemented ESP package that allows analysis of timed Petri nets with PH distributed firing times using phase-type expansions. Distribution of completion time can be obtained. Further numerical examples of computing performability measures using ESP are in [20].

We have implemented a front-end package to SHARPE called GSHARPE [15], which permits analysis of a class of non-Markovian reward models (where after a change of state in the reliability model, the system cannot keep track of the past and any task being executed on the system must be restarted – the preemptive repeat different policy). The non-Markovian (reward) model is converted to a Markov (reward) model using phase approximations which is then solved by SHARPE. All the other features of SHARPE are supported and performability analysis as supported by SHARPE can be carried out.

- Analytic numeric solution of the non-Markovian model. Recently there have been significant new developments in this area. Ajmone Marsan and Chiola [1] analyzed stochastic Petri nets with deterministic and exponentially distributed firing times which have come to be known as DSPNs (deterministic and stochastic Petri nets). Linde mann [13] has designed a software package called DSPNexpress that allows steady-state analysis of DSPNs under certain restrictive conditions. Choi et al [6] have proposed a more general class of timed Petri nets known as Markov regenerative stochastic Petri nets (MRSPNs). These are analyzed by Markov regenerative processes under certain restrictive conditions. Both the transient and steady-state solutions can be obtained numerically. Although none of these studies discuss reward based measures, reward rates may be associated with markings of the net and reward-based performability measures can be computed.

References


