Proceedings of the 1990 Winter Simulation Conference
Osman Balci, Randall P. Sadowski, Richard E. Nance (eds.)

ANALYSIS OF MARKOV CHAINS USING SIMULATION GRAPH MODELS

Enver Yücesan

INSEAD
European Institute of Business Administration
Boulevard de Constance
77305 Fontainebleau Cedex, France

ABSTRACT

Markov chain analysis is a valuable tool in studying a wide range of problems. Usefully detailed models, however, are usually too cumbersome to be represented as Markov chains. In this paper, we demonstrate the construction of Simulation Graph Models of Markov chains. This approach enables one to evaluate the chain either numerically through simulation or analytically through path analysis. This method can be used in conjunction with simulation to address such problems as rare event estimation, initialization bias, and determination of initial conditions.

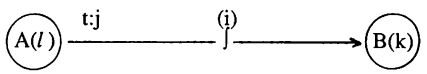
1. INTRODUCTION

Stochastic models are valuable in analyzing a wide range of problems. However, usefully detailed stochastic models are usually so complex that obtaining an analytic solution is extremely difficult or even impossible. Discrete event simulation is usually the only practical method for studying the behavior of such models [Shedler 1987]. Representing the system as a continuous time Markov chain is difficult or impractical because of the amount of detail that must be included in the definition of a state and/or simply because of the sheer number of states that result from such a definition. In fact, Fox [1987] argues that, when it is possible to produce the associated transition matrix -or, equivalently, the chain's generator- in its entirety, deterministic numerical methods are more efficient. Simulation, on the other hand, becomes increasingly attractive as the number of states grows.

One possible solution to this problem is a procedure by Fox [1987] where the rows of the transition matrix are produecd only as needed, generating the transitions directly. An alternative approach is presented in this paper, where a finite representation of the stochastic process is obtained even when the chain's generator or the transition matrix is not finite. This is achieved by building a Simulation Graph Model of the Markov chain. This approach enables the user either to directly simulate the associated stochastic process and obtain estimates of the desired measures of performance, or to analyze the directed paths in the Simulation Graph Model and analytically compute the probabilities of possible realizations.

2. SIMULATION GRAPH MODELS

A *Simulation Graph* is a structure of the elements in a discrete event system that facilitates the developments of correct simulation models. Events are represented as vertices on the graph. Each event vertex is associated with a set of changes to state variables. Relationships between events, on the other hand, are represented as directed edges between pairs of vertices. Each edge depicts under what conditions and after how long of a time delay an event can schedule or cancel another event. More specifically,



indicates that "t time units after the occurrence of event A, event B is scheduled to occur, with parameter string k := j, provided that condition (i) holds at the time event A occurs." The parameter string carries information pertaining to a particular event instance. These strings can be passed in a model through vertex and edge attributes. When the origination vertex of an edge is executed, the expressions in the edge's attribute list are evaluated. When the destination vertex is subsequently executed, the state variables in its attribute list take on the values that have been computed for the expressions in the scheduling edge's attribute list. The main value of these lists is in defining the particular system entities to which an event pertains. A complete treatment of Simulation Graph Models is presented in [Yucesan and Schruben 1989].

3. SIMULATION GRAPH REPRESENTATION OF MARKOV CHAINS

In this section, the construction of Simulation Graph Models of Markov chains is demonstrated. This construction enables the numerical evaluation (simulation) of the underlying stochastic process to obtain estimates of the desired measures of performance.

In this representation, each vertex corresponds to one particular *state* of the process. Edge conditions are used to determine the next state. For discrete time Markov chains, the edge delay times are all taken to be unity. For continuous time Markov chains, the edge delay times are sampled from the sojourn time distribution. The construction is illustrated next with a series of examples.

Example: Discrete Time Markov Chains

Suppose that the following transition matrix, P, is associated with a discrete time Markov chain.

$$P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & \alpha & 1-\alpha \\ 0 & 1-\beta & \beta \end{bmatrix}$$

The associated Simulation Graph Model is depicted in Figure 1. The state variables used in this model are:
X - current value of the process,
p - transition probability,
q - transition probability.
The edge conditions are:
(i) $p \le \alpha$,
(ii) $p > \alpha$,
(iii) $q \le \beta$,
(iv) $q > \beta$.
The edge delay times (not shown on Figure 1) are all taken to be equal to one. The event descriptions are presented in Table 1.

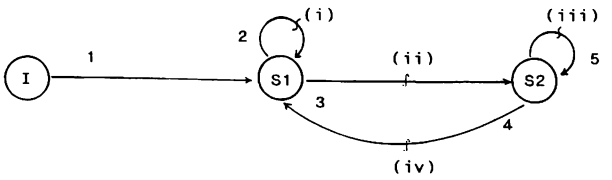


Figure 1. Discrete Time Markov Chain

Table 1. Event Descriptions for Discrete Time Markov Chains

Event Type	Event Description	State Changes
I	Initialization	X := 0 p := 0 q := 0
S1	Process in State 1	X := 1 p := U[0,1]
S2	Process in State 2	X := 2 q := U[0,1]

Note that depending on the desired measure of performance (e.g., steady state probabilities, first-passage times), other performance-monitoring state variables can be added to the model.

Example: Continuous Time Markov Chains

Consider a capacitated Markovian single-sever queue (M/M/1/N). For exposition purposes, let N = 2. Also let λ represent the arrival rate and μ represent the service rate. Hence, the state variables used in the model are:
X - number of customers in the system,
λ - arrival rate of customers,
μ - service rate,
p - transition probability.
This is a birth and death process [Taylor and Karlin 1984]. The process' generator, Q, is given by

$$Q = \begin{bmatrix} -\lambda & \lambda & 0 \\ \mu & -(\lambda+\mu) & \lambda \\ 0 & \mu & -\mu \end{bmatrix}$$

The Simulation Graph Model is depicted in Figure 2. Note that
$t_1 \sim \exp(\lambda)$,
t_2 and $t_3 \sim \exp(\lambda+\mu)$,
$t_4 \sim \exp(\mu)$.
The edge conditions are given by:
(i) $p \leq \mu/(\lambda+\mu)$,
(ii) $p > \mu/(\lambda+\mu)$.
The event descriptions are presented in Table 2.

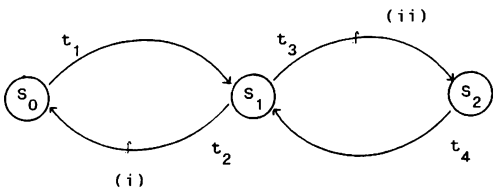


Figure 2. Continuous Time Markov Chain

Table 2. Event Descriptions for Continuous Time Markov Chains

Event Type	Event Description	State Changes
S_0	State 0	X := 0
S_1	State 1	X := 1 p := U[0,1]
S_2	State 2	X := 2

Again, performance monitoring state variables can be added to the model depending on the objectives of the study.

Example: Markov Chains with Countable State Space

Consider now an M/M/1 queue with no capacity constraints ($N=\infty$). The state, X, depicts the number of customers in the system. The generator, Q, is then given by

$$Q = \begin{bmatrix} -\lambda & \lambda & 0 & 0 & .. \\ \mu & -(\lambda+\mu) & \lambda & 0 & .. \\ 0 & \mu & -(\lambda+\mu) & \lambda & .. \\ . & . & . & . & .. \end{bmatrix}$$

The model is presented in Figure 3. The initialization vertex, INIT, establishes the initial values of the state variables, which were defined in the previous example. The vertex ST depicts the current state of the process. The information about the next state is carried by the edge and vertex attributes. Three contingencies exist in the model: (1) when the system is empty, the only possible transition is triggered by the arrival of a new customer, with interarrival time $t_0 \sim \exp(\lambda)$; (2) when the system is in state n, the process moves to state (n+1) with probability $\lambda/(\lambda+\mu)$, or (3) from state n to state (n-1) with probability $\mu/(\lambda+\mu)$ after a time delay of $t_1 \sim \exp(\lambda+\mu)$. The edge conditions are:
(i) $p \leq \lambda/(\lambda+\mu)$ & $X \neq 0$,
(ii) $p > \lambda/(\lambda+\mu)$ & $X \neq 0$,
(iii) $X = 0$.

The event descriptions are presented in Table 3.

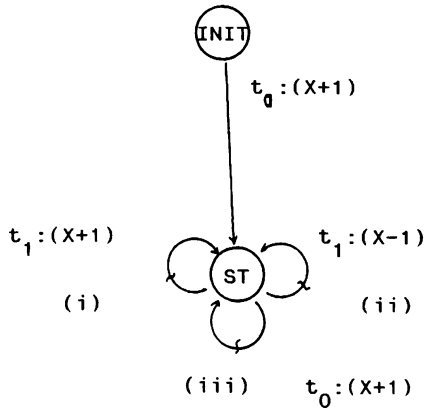


Figure 3. Markov Chain with Countable State Space

Table 3. Event Descriptions for Markov Chains with Countable State Space

Event Type	Event Description	State Changes
INIT	Initialization	X := 0 p := 0 λ := a μ := b
ST	State of System	X := I p := U[0,1]

Performance monitoring variables can be added to the model in accordance with the desired measure of performance.

4. SAMPLE PATH ANALYSIS

An efficient alternative to numerically evaluating (simulating) the implementation of Simulation Graph Models of Markov chains is to analyze the directed paths of the graph for computing probabilities associated with possible realizations. In this section, this procedure is illustrated after some preliminary definitions.

4.1 Basic Control Paths

Following McCabe [1976], a *program control graph* is defined as follows: a directed graph with unique entry and exit vertices is associated with a computer program. Each vertex on the graph corresponds to a block of code in the program where the flow is sequential, and the edges correspond to branches taken in the program.

Theorem: In a strongly connected graph, G, the cyclomatic number, $\eta(G)$, is equal to the maximum number of linearly independent directed cycles.

McCabe uses this theorem along with the program control graph to define the *basic control paths* in a program. The latter is a set of paths in the control graph whose linear combinations represent all possible paths through the control graph and, hence, the program. Figure 4 depicts an example with $\eta(G)$=e-n+p = 6-4+1=3, where G is a graph with e edges, n vertices and p connected components. Note that edge 6 is added to the graph to make it strongly connected. One can then choose the following set of independent directed cycles to form a basis:

B_1: (a b d a), (b c b), (a c b d a).

A row vector is associated with every element of the basis, B_1:

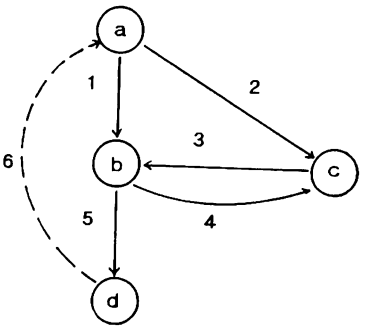


Figure 4. Program Control Graph

edges	1	2	3	4	5	6
a b d a	1	0	0	0	1	1
b c b	0	0	1	1	0	0
a c b d a	0	1	1	0	1	1

For instance, the path (a (b c)^2 b d a) corresponds to the vector (1 0 2 2 1 1), and the vector addition of (a b a d) + 2(b c b) yields the desired result.

4.2 Extension to Simulation Graphs

Simulation Graphs are analogous to program control graphs. On a Simulation Graph, each vertex essentially corresponds to a block of code in the program where the flow is sequential. The edges, on the other hand, are analogous to branches taken in a program. Without loss of generality, we can assume that Simulation Graphs have unique entry and exit vertices. An initialization vertex, where the initial state of the model is established, is the entry vertex. The exit vertex, on the other hand, is a special vertex on the Simulation Graph, which terminates the execution of the simulation run upon the satisfaction of the termination conditions (a so-called *end vertex*).

With this correspondance in mind, the results of the previous section can be applied to the Simulation Graph Models of Markov chains. Once a basic set of control paths is determined for the Simulation Graph, all possible sample paths can be expressed as linear combination of these paths. In addition, it is possible to associate a probability measure with each of the basic paths; and since these paths are independent, the probabilities can be combined to compute the likelihood of observing a particular realization.

More specifically, let $\pi_1, \pi_2, \ldots, \pi_k$ be the basic control paths of a Simulation Graph Model. Also, let p_i be the probability of observing the i^{th} basic path (i=1,2,3,...,k). Then, if a particular realization can be expressed as the following linear combination:

$$S = \omega_1\pi_1 + \omega_2\pi_2 + \ldots + \omega_k\pi_k$$

then, by the independence of these paths, the probability of observing such a realization is given by:

$$P[S] = p_1^{\omega_1} * p_2^{\omega_2} * \ldots * p_k^{\omega_k}.$$

Example: Discrete Time Markov Chains

Consider the Simulation Graph Model of the discrete time Markov chain of Figure 1 with $\eta(G)$=4. A set of basic control paths is given by:

B: (I S1 S2), (I S1 S1 S2), (I S1 S2 S1 S2), (I S1 S2 S2).

As a vector, they can be represented as:

	1	2	3	4	5
π_1: I S1 S2	1	0	1	0	0
π_2: I S1 S1 S2	1	1	1	0	0
π_3: I S1 S2 S1 S2	1	0	2	1	0
π_4: I S1 S2 S2	1	0	1	0	1

The associated probabilities are given by:

$p_1 = 1-\alpha$,

$p_2 = \alpha(1-\alpha)$,

$p_3 = \beta(1-\alpha)$,

$p_4 = (1-\alpha)^2(1-\beta)$.

Suppose we are interested in observing the following sample path: (I a^4 b). This path can be represented as $-3\pi_1 + 3\pi_2 + \pi_4$. Hence, it can be observed with the following probability:

$$P[S] = p_1^{-3} p_2^3 p_4$$
$$= (1-\alpha)^{-3} \alpha^3 (1-\alpha)^3 (1-\alpha)^2 (1-\beta)$$
$$= \alpha^3 (1-\alpha)^2 (1-\beta).$$

Example: Continuous Time Markov Chains

Consider the Simulation Graph Model of Figure 2, where $\eta(G)=3$. A set of basic control paths is given by:
B: (S0 S1 S2), (S0 S1 S0 S1 S2), (S0 S1 S2 S1 S2).
As a vector they can be represented as:

	1	2	3	4	p_i
π_1: S0 S1 S2	1	0	1	0	p
π_2: S0 S1 S0 S1 S2	2	1	1	0	pq
π_3: S0 S1 S2 S1 S2	1	0	2	1	p^2

where, $p = \lambda/(\lambda+\mu)$ and $q = 1-p$. Any particular sample path can then be expressed as a linear combination of these basic paths, and its probability can be computed accordingly.

Example: Markov Chains with Countable State Space

Consider a general birth and death process. Let λ_i be the birth parameter and μ_i be the death parameter. The process goes from state i to state i+1 with probability $p_i = \lambda_i /(\lambda_i + \mu_i)$, and from state i to state i-1 with probability $q_i = 1 - p_i$. The sojourn time at each state is given by $t \sim \exp(\lambda_i+\mu_i)$. The Simulation Graph Model of the general birth and death process is presented in Figure 5. Since $\eta(G)=3$, the basic control paths are given by:

	1	2	3	p_i
π_1: (INIT) (I)	1	0	0	1
π_2: (INIT) (I) (I+1)	1	1	0	p_i
π_3: (INIT) (I) (I-1)	1	0	1	q_i

Then, the sample path (INIT, 0, 1, 2, 1, 2, 3) can be represented by $S: -4\pi_1 + 4\pi_2 + \pi_3$. Then,

$$P[S] = 1^{-4} p_i^4 q_i$$
$$= p_0 p_1^2 p_2 q_2.$$

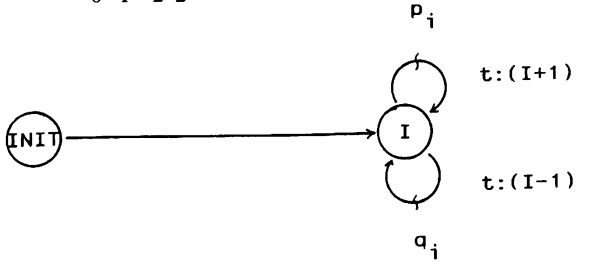


Figure 5. General Birth and Death Process

5. APPLICATIONS OF SAMPLE PATH ANALYSIS

The information about sample paths can be used independently or in conjunction with the simulation run to improve the precision of estimators for desired measures of performance. Several possible applications are highlighted below.

Analysis of the Chain

Some popular measures of performance can be calculated through the analysis of basic paths. For example, in models with absorbing states, the probability of absorption or mean time until absorption starting at a particular state can be computed by using the probabilities associated with these paths.

Estimation of Rare Events

The analysis of directed paths in the Simulation Graph can be used in computing rare event probabilities *without* even running the simulation. This is achieved by computing the probabilities associated with different sample paths that lead into that state which would result from the execution of the rare event.

Possible Test for Initialization Bias

An initialization test can be developed based on the likelihood of the observed sample path. A control mechanism can be imposed on the simulation, that, at given intervals, computes the probability of observing the path generated thus far. Since the probabilities of all possible sample paths can be computed, the observation of a highly unusual (i.e., highly unlikely) path would be a sign of initialization bias.

Initialization of a Run

Since probabilities of different sample paths can be computed, this information can be used to start a simulation run in a "typical" state. Alternatively, the initial state can be chosen at random based on the sample path probabilities.

6. CONCLUDING COMMENTS

The representation of Markov chains through Simulation Graph Models provides a practical way to analyze these models either analytically or numerically. However, analytic methods can be used in conjunction with simulation to enhance the precision of estimators. The next task is to extend these procedures to non-Markovian models.

ACKNOWLEDGEMENT

The author wishes to thank Professor Lee Schruben for his comments during earlier discussions on the topic.

REFERENCES

Fox, B.L. (1987), "Beating the Future Event Schedules," Technical Report No.761, School of OR&IE, Cornell University, Ithaca, NY.

McCabe,T.J.(1976), "A Complexity Measure," *IEEE Transactions on Software Engineering, SE-2,* 4, 308-320

Shedler, G.S. (1987), *Regeneration and Networks of Queues,* Springer-Verlag, New York, NY.

Taylor, H.M. and S. Karlin (1984), *An Introduction to Stochastic Modeling,* Academic Press, Orlando, FL.

Yücesan, E. and L.W. Schruben (1989), "Simulation Graphs for the Design and Analysis of Discrete Event Simulation Models," Working Paper Series No.89/60/TM, INSEAD, Fontainebleau, France.