Universidade Federal de Pernambuco
Centro de Informática

Graduação em Engenharia da Computação

Modelagem de Sistemas Computacionais utilizando o formalismo matemático de Cadeias de Markov

Lucas Aranha Barreto

Recife
Novembro de 2009

Universidade Federal de Pernambuco
Centro de Informática

Lucas Aranha Barreto

Modelagem de Sistemas Computacionais utilizando o formalismo matemático de Cadeias de Markov

Trabalho apresentado ao Curso de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Engenheiro da Computação.

Orientador: Marcília Andrade Campos

Recife
Novembro de 2009

Agradecimentos

Primeiramente, agradeço à toda minha família, especialmente aos meus pais, Antônio Carlos e Maria Lúcia, por serem pessoas extremamente especiais. Eu devo a eles a oportunidade de fazer esse curso, a formação do meu caráter e o meu jeito de ser, além de todo o resto. Também agradeço à minha irmã, Fernanda, que é uma pessoa fantástica e que sempre me inspirou por ter tantas qualidades. À minha avó, Dadinha, pelo carinho e cuidado de sempre, pois participa da minha vida desde que eu nasci. Aos meus sobrinhos, Gui e Rafa, com quem me divirto muito, sempre esperando ser um bom exemplo para eles. À minha tia Vanda, que apesar de não ser tia de sangue, é como se fosse, e é meu maior exemplo de superação de dificuldades impostas pela vida. Ao meu cunhado, Telmo, gente boa, tinha que ser pra estar com minha irmã, claro. À Letícia, minha sobrinha, que ainda está na barriga da mãe, mas merece agradecimentos porque já me faz feliz mesmo estando escondida ainda.

Preciso agradecer também à minha namorada, Ju, que é uma pessoa única e me faz feliz por seu jeito de ser e de gostar de mim. Aos meus amigos, fundamentais na minha vida, tanto os que fiz em Recife como os que tenho desde que morava em Aracaju, minha cidade. Dos meus amigos de Aracaju, devo destacar Moab, Andrius, Bruno (vulgo Vaquinha), Rodrigo (vulgo Gago), Marcelo (vulgo Motor) e Jerônimo, que veio comigo para Recife e ainda hoje é um grande amigo meu. Dos que conheci em Recife, devo destacar Diego (vulgo Chucky), Rodrigo (vulgo Reaf), Antônio (vulgo Espanta), Bruno (vulgo Apebão), Heitor, Pablo, Anderson (vulgo Cabelinho), Eduardo (vulgo Índio), Rafael Borba, Diego Riff, André (vulgo Mamão), Lucas (vulgo Bel), Felipe, Júnior, Denyson, Bruno (vulgo Baby), André, Bruno (vulgo Bruno Verde), Dani e Arnaldo.

Todas as pessoas acima, desde minha família até o último amigo citado, me influenciaram e me influenciam de uma maneira ou de outra na minha vida. Acredito que as pessoas devem sempre estar atentas aos que lhe rodeiam, assim podendo absorver todos os ensinamentos possíveis deles, intencionais ou não, dos mais simples até os mais complexos. Vivo assim, aprendendo com os outros e tentando ensinar também, sempre procurando melhorar como pessoa.

A todos, muito obrigado, de coração.

Resumo

Os profissionais responsáveis pelo desenvolvimento de sistemas computacionais muitas vezes têm o interesse de prever o comportamento do sistema a ser desenvolvido antes da implementação do mesmo. Isso acontece porque testar e alterar um sistema já construído são tarefas custosas. Uma maneira de minimizar esses custos é criar um modelo matemático que imita o sistema e viabiliza a análise de características comportamentais do mesmo em diversas situações, em critérios como performance, confiabilidade e disponibilidade.

Esse trabalho tem o objetivo de descrever o processo de modelagem e análise de sistemas computacionais utilizando o formalismo matemático de Cadeias de Markov. Os sistemas que podem ser representados como um conjunto de estados e transições, e além disso se comportam de acordo com a propriedade de Markov, são os sistemas qualificados para serem modelados por Cadeias de Markov.

Depois do modelo ser concluído, a análise pode ser feita por soluções analíticas ou por soluções de simulação. Esses dois tipos de solução têm suas vantagens e desvantagens, devendo ser escolhidas dependendo do contexto. Várias medidas de desempenho podem ser inferidas no processo de análise, revelando características dinâmicas do sistema, o que é muito útil para os desenvolvedores. Esse trabalho mostra exemplos de modelos e soluções analíticas e de simulação, com o intuito de melhorar a compreensão do processo de modelagem e análise por Cadeias de Markov, assim fornecendo a base de conhecimento necessária para o uso dessa técnica.

Abstract

Individuals responsible for development of computer systems often have the interest to predict the behavior of the system to be developed before implementation. This is because test and change the system already built are expensive tasks. One way to minimize these costs is to create a mathematical model that mimics the system and enables the analysis of behavioral characteristics of it in many situations on criteria such as performance, reliability and availability.

This paper aims to describe the process of modeling and analysis of computer systems using the mathematical formalism of Markov chains. Systems that can be represented as a set of states and transitions, and also behave according to the Markov property are the systems eligible to be modeled by Markov chains.

After the model is completed, the analysis can be done by analytical solutions or simulation solutions. These two types of solutions have their advantages and disadvantages, and should be chosen depending on the context. Several performance measures can be inferred in the review process, revealing dynamic characteristics of the system, which is very useful for developers. This study describes examples of models and analytical solutions and simulation in order to improve the understanding of process modeling and analysis using Markov chains, thus providing the knowledge base required to use this technique.

Sumário

1 Introdução

1.1 Motivação

Os desenvolvedores de sistemas computacionais muitas vezes precisam prever o comportamento desses sistemas em funcionamento antes de construí-los. Testar e corrigir um sistema somente depois do mesmo estar implementado é uma tarefa custosa. É muito importante conhecer características de confiabilidade e desempenho do sistema que vai ser desenvolvido. Para avaliar o comportamento de um sistema antes de desenvolvê-lo é útil fazer um modelo desse sistema. Esse modelo é uma abstração que permite simular o comportamento do sistema através de parâmetros conhecidos pelos desenvolvedores.

No caso da computação, muitos sistemas podem ser representados por um conjunto de estados e transições entre esses estados. Outra característica dos sistemas computacionais é a presença de variáveis não-determinísticas, ou seja, o sistema se comporta de maneira aleatória, existe uma probabilidade do sistema passar de um estado para outro, ao invés de uma condição exata. Assim, os modelos úteis para prever o comportamento desses sistemas são os modelos que representam processos estocásticos, que são processos que evoluem no tempo probabilisticamente.

A principal teoria para modelar sistemas que podem ser descritos por um diagrama de estados e nos quais as transições entre esses estados dependem de probabilidades é a teoria das Cadeias de Markov.

As Cadeias de Markov tem esse nome porque foram introduzidas pelo matemático Andrei Andreyevich Markov, em 1907. Markov nasceu em 14 de junho de 1856 em Ryazan, na Rússia, graduou-se na Universidade de São Petersburgo em 1878 e se tornou professor da mesma em 1886. Por volta de

1900 começou a estudar processos estocásticos, o que resultou na descrição do processo estocástico conhecido como Processo de Markov. As Cadeias de Markov são usadas para análise de desempenho desde 1950, aproximadamente. Como as cadeias de Markov são usadas há muito tempo, existem várias técnicas de solução numérica, simulação, métodos de geração de espaço de estados e softwares que auxiliam no processo de modelagem e avaliação. [3]

Andrei Andreyevich Markov (1856-1922)

1.2 Estrutura do documento

Neste trabalho será feita a descrição de como modelar sistemas computacionais utilizando o formalismo matemático de cadeias de Markov, teoria útil para modelar sistemas computacionais que podem ser representados por diagramas de estados.

Nas próximas seções desse capítulo será mostrada a abordagem geral para modelagem e avaliação de um sistema computacional, destacando as propriedades de desempenho que podem ser verificadas. Também há uma seção de revisão de conceitos relevantes de probabilidade.

Os capítulos seguintes são focados na definição das Cadeias de Markov e no processo de modelagem e avaliação de sistemas. Para enriquecer a explicação serão mostrados exemplos de modelagem de sistemas simples, inclusive com o auxílio de um software de modelagem existente.

1.3 Processo de modelagem

A primeira etapa do processo de modelagem é a formalização, onde o desenvolvedor do modelo gera uma descrição formal do sistema real. Partindo de uma descrição informal do sistema, em linguagem natural, por exemplo, o desenvolvedor cria o modelo formal para o sistema, usando uma teoria específica. Essa teoria permite ao desenvolvedor obter uma visão abstrata e simplificada do sistema real.[3],[6]

A teoria abordada nesse trabalho é a teoria de Cadeias de Markov, que é utilizada para modelar sistemas que podem ser representados por diagramas de estados. A modelagem por Cadeias de Markov tem o foco no comportamento dinâmico do sistema, visando simular as situações possíveis e os efeitos do funcionamento do sistema real. Na Figura 1.1 é mostrado o processo de formalização.

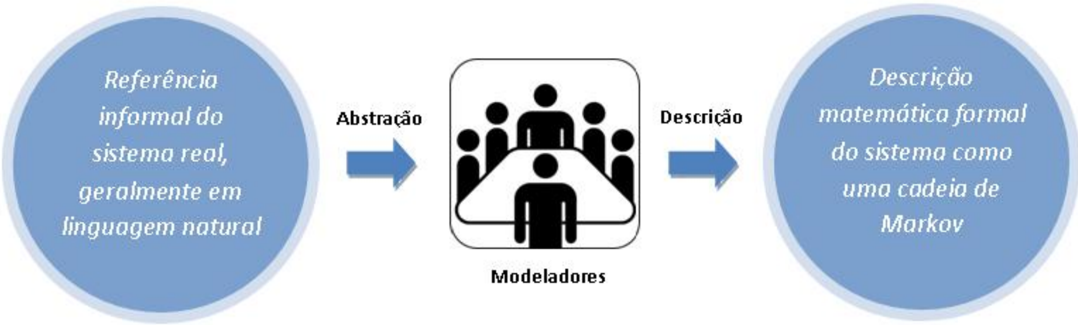


Figura 1.1 Processo de formalização

As principais abstrações que devem ser feitas para construir o modelo são:[2]

- **Fazer a associação das durações das atividades do sistema com variáveis randômicas**. Exemplo: Um servidor de um sistema de rede recebe em média uma requisição a cada 10 segundos. Este servidor consegue armazenar no máximo 3 requisições. As requisições levam em média 5 segundos para serem atendidas.

- **Definir as probabilidades de transição entre os estados do sistema**. Exemplo: Dois processadores rodam em paralelo, os dois funcionando, esse é o estado A. Um dos processadores está rodando e o outro está com problemas e não está funcionando, esse é estado B. A probabilidade de um dos processadores parar de funcionar é de 1% em cada intervalo de tempo. Essa probabilidade define uma transição do estado A para o estado B.

Para criar e avaliar o modelo é necessário definir uma escala de tempo. Os eventos do sistema devem ser medidos por um intervalo de tempo determinado, por exemplo, a cada hora.

Depois do modelo formal ser definido, é possível "pegar"um determinado momento durante a operação do sistema e saber as probabilidades de evolução do sistema para outros estados nos instantes de tempo seguintes.

Dependendo do tipo de situação, a cadeia de Markov pode ser de tempo contínuo (CTMC - Continuos Time Markov Chain) ou de tempo discreto (DTMC - Discrete Time Markov Chain). Nas situações em que o sistema pode mudar de estado a qualquer momento deve ser usada a CTMC, caso contrário deve ser usada a DTMC. Os tipos de cadeias de Markov vão ser detalhados no Capítulo 2.

1.4 Avaliação do modelo

Após a construção do modelo formal do sistema, a avaliação do mesmo com relação às características de desempenho e confiabilidade deve ser realizada. Para isso existem diferentes métodos de avaliação de modelos. A escolha do método depende das abstrações feitas no processo de formalização. As soluções mais utilizadas são:[2]

Soluções analíticas: As soluções analíticas se baseiam em representar o modelo formalizado como uma equação ou como um conjunto de equações. Dessa equação ou conjunto de equações podem ser obtidas informações relevantes sobre o sistema através de soluções de forma fechada (closed-form solutions) ou de medidas aproximadas calculadas por algoritmos de matemática numérica.

Soluções de forma fechada podem ser obtidas se a cadeia de Markov gerada não for muito grande ou se tiver uma estrutura regular. A vantagem das soluções de forma fechada é a rapidez do cálculo das medidas de performance do sistema, por causa da menor complexidade. Quando não é possível encontrar uma solução de forma fechada é preciso encontrar a solução do modelo utilizando algoritmos de matemática numérica, que obtêm resultados aproximados. Existem vários algoritmos desse tipo, e a desvantagem deles é que geralmente são mais complexos, retardando a solução do modelo.

Soluções de simulação: Em alguns casos, não é possível encontrar uma solução analítica para o modelo, isso porque não se conhece um método para derivar equações daquele modelo, ou porque a complexidade de computação de um algoritmo numérico para esse modelo é muito alta. Nesse caso é feita uma simulação do modelo. Ao invés de resolver equações derivadas do modelo, um determinado algoritmo executa o modelo e recolhe as informações do comportamento observado durante a simulação, com o objetivo de obter as medidas de desempenho do sistema.

Muitas vezes o método de simulação é melhor que o de soluções analíticas, mesmo quando as mesmas existem para determinado sistema. Isso acontece porque a simulação é flexível, podendo ser executada várias vezes para refinar os resultados, além de ser computacionalmente mais viável nos casos em que o modelo contém grande número de estados e soluções numéricas de difícil resolução e interpretação.

1.5 Propriedades de desempenho

O objetivo de modelar um sistema é prever seu comportamento dinâmico, ou seja, em funcionamento. O modelo sob avaliação deve fornecer certas características do sistema que são relevantes para os desenvolvedores, as quais são, principalmente:[4]

Performance: O grau em que o sistema executa suas funções em critérios como velocidade, precisão e uso de memória, por exemplo.

Confiabilidade: A probabilidade de que o sistema não vai falhar por um dado tempo sob condições especificadas.

Disponibilidade: A capacidade do sistema de executar uma função requerida em um determinado instante ou período de tempo.

1.6 Conceitos de probabilidade

Antes de explicar a teoria de Cadeias de Markov e o uso da mesma para modelar sistemas computacionais, é interessante fornecer o conhecimento básico necessário para o entendimento de todo o processo de formalização matemática e criação do modelo.

Nesta seção serão abordadas definições básicas da teoria da probabilidade.[2],[5]

1.6.1 Definições

- **Fenômeno aleatório**

  É um fenômeno imprevisível, não é possível saber quando irá acontecer no futuro. Ao lançar um dado, por exemplo, não podemos saber qual a face resultante antes de jogarmos o dado, só podemos saber a **probabilidade** de determinada face ser a resultante.

- **Evento discreto**

  Um evento é discreto quando sua ocorrência só é verificada em determinados instantes de tempo dentro de um período. Por exemplo, a cada minuto ou a cada hora.

- **Evento contínuo**

  Um evento é contínuo quando pode ocorrer em qualquer instante de tempo.

- **Variável aleatória**

  Uma variável aleatória representa o resultado de um experimento aleatório. Podemos saber somente a probabilidade da variável assumir um certo valor dentro de um conjunto de valores possíveis.

- **Distribuição de probabilidade**

  Para estudar uma variável aleatória é interessante identificar os valores que a mesma pode assumir e a frequência de ocorrência de cada valor. A função que representa essa frequência é chamada de distribuição de probabilidade.

- **Memoryless**

  Em um sistema denominado "memoryless", saber o que aconteceu no passado não ajuda a prever o que acontecerá no futuro. Quando uma moeda é lançada, por exemplo, mesmo tendo jogado a moeda várias vezes anteriormente, os resultados anteriores não influenciam de modo algum o próximo resultado. Assim, em sistemas "memoryless", dado o estado atual, o próximo estado do sistema não depende dos estados passados.

1.6.2 Variáveis aleatórias

Nos sistemas computacionais existem inúmeras variáveis aleatórias, como a quantidade de pacotes que chega a um roteador no período de uma hora ou o tempo entre a chegada de duas tarefas consecutivas a um sistema. Nota-se que no primeiro exemplo a variável só pode assumir valores discretos, pois ela representa o número de pacotes. Entretanto, no segundo exemplo, a variável em questão representa o tempo entre a chegada de duas tarefas, assim esse valor pode assumir valores contínuos. Por isso existe uma classificação específica para o tipo de variável aleatória, que divide as variáveis aleatórias em discretas e contínuas.

1.6.1.1 Variáveis aleatórias discretas

Uma variável aleatória discreta somente pode assumir valores discretos. A variável é descrita pelos valores possíveis que ela pode ter e pelas probabilidades de ocorrência dos mesmos. O conjunto de probabilidades de ocorrência de cada valor é chamado de **função de probabilidade** da variável. Para uma variável aleatória discreta X, assumindo que X não assume valores negativos, a função de probabilidade é definida como:

$$p_k = P(X = k) \quad \text{para} \quad \text{k = 0, 1, 2...,} \tag{1}$$

que representa a probabilidade da variável X assumir um valor k. É necessário que:

$$\begin{aligned} P(X = k) &\geq 0 \qquad (2) \\ \sum_k P(X = k) &= 1 \qquad (3) \end{aligned}$$

Vários parâmetros importantes podem ser derivados das variáveis aleatórias discretas:

- **Valor médio ou valor esperado**

$$\overline{X} = E(X) = \sum_k k \cdot P(X = k) \tag{4}$$

O valor esperado representa o valor médio de uma experiência se ela for repetida muitas vezes.

- **Variância**

$$\sigma_X^2 = V(X) = E(X^2) - (E(X))^2 \tag{5}$$

onde σ_X é chamado de desvio padrão.

A variância de uma variável aleatória é uma medida da sua dispersão, que indica a distância geral entre os seus valores e o valor esperado.

Existem algumas variáveis aleatórias discretas muito importantes, como a de Bernoulli, a binomial, a geométrica e a de Poisson, que descrevem tipos de eventos específicos e que ocorrem frequentemente em várias situações reais.

1.6.1.2 Variáveis aleatórias contínuas

Uma variável aleatória contínua X pode assumir qualquer valor no intervalo [a,b], onde $-\infty \leq a < b \leq \infty$. Ela é descrita pela função de distribuição acumulativa, ou FDA:

$$F_X(x) = P(X \leq x), \tag{6}$$

que especifica a probabilidade de que a variável X tome um valor menor ou igual à x.

A **função densidade de probabilidade (FDP)** $f_X(x)$ pode ser usada ao invés da função de distribuição acumulada:

$$f_X(x) = \frac{dF_X(x)}{dx} \tag{7}$$

Algumas propriedades da FDP são:

$$f_X(x) \geq 0 \quad \text{para todo x}, \tag{8}$$

$$\int_{-\infty}^{\infty} f_X(x)dx = 1, \tag{9}$$

$$P(x_1 \leq X \leq x_2) = \int_{x_1}^{x_2} f_X(x)dx, \tag{10}$$

$$P(X = x) = \int_{x}^{x} f_X(x)dx = 0, \tag{11}$$

$$P(X > x_3) = \int_{x_3}^{\infty} f_X(x)dx = 0, \tag{12}$$

A função densidade de probabilidade é análoga à função de probabilidade das variáveis aleatórias discretas.

- **Valor médio ou valor esperado**

$$\overline{X} = E(X) = \int_{-\infty}^{\infty} x \cdot f_X(x)dx \tag{13}$$

- **Variância**

$$\sigma_X^2 = V(X) = E(X^2) - (E(X))^2, \tag{14}$$

onde σ_X é chamado de desvio padrão.

Existem algumas variáveis aleatórias contínuas muito importantes, como a normal, a exponencial, a hiperexponencial e a gama. Essas variáveis descrevem eventos contínuos muito frequentes em situações reais.

1.6.3 Variáveis aleatórias n-dimensionais

A **função de probabilidade conjunta** de variáveis aleatórias discretas $X_1, X_2 ..., X_n$ é dada por:

$$p(x_1, x_2, ..., x_n) = P(X_1 = x_1, X_2 = x_2, ..., X_n = x_n) \tag{15}$$

e representa a probabilidade de $X_1 = x_1, X_2 = x_2, ...X_n = x_n$. Para as variáveis aleatórias contínuas, temos a **função de distribuição conjunta**:

$$F_X(x) = P(X_1 \leq x_1, X_2 \leq x_2, ..., X_n \leq x_n) \tag{16}$$

que representa a probabilidade de $X_1 \leq x_1, X_2 \leq x_2, ..., X_n \leq x_n$, onde $X = (X_1, X_2, ..., X_n)$ é a variável aleatória n-dimensional e $x = (x_1, x_2, ..., x_n)$.

1.6.3.1 Independência entre variáveis aleatórias

X_1 e X_2 são variáveis aleatórias independentes quando o resultado de X_1 não influencia o resultado de X_2 nem o resultado de X_2 influencia o resultado de X_1.

Variáveis aleatórias $X_1, X_2, ..., X_n$ são chamadas de estatisticamente independentes se, para o caso discreto:

$$P(X_1 = x_1, X_2 = x_2, ..., X_n = x_n) = P(X_1 = x_1) \cdot P(X_2 = x_2) \cdot ... \cdot P(X_n = x_n) \tag{17}$$

e para o caso contínuo:

$$P(X_1 \leq x_1, X_2 \leq x_2, ..., X_n \leq x_n) = P(X_1 \leq x_1) \cdot P(X_2 \leq x_2) \cdot ... \cdot P(X_n \leq x_n) \tag{18}$$

Se isso não for verdade, as variáveis são estatisticamente dependentes.

1.6.3.2 Probabilidade condicional

A probabilidade condicional

$$P(X_1 = x_1 \mid X_2 = x_2, ..., X_n = x_n) \tag{19}$$

é a probabilidade de $X_1 = x_1$ dado que $X_2 = x_2, ..., X_n = x_n$. Temos que, para o caso discreto:

$$P(X_1 = x_1 \mid X_2 = x_2, ..., X_n = x_n) = \frac{P(X_1 = x_1, X_2 = x_2, ..., X_n = x_n)}{P(X_2 = x_2, ..., X_n = x_n)} \tag{20}$$

e para o caso contínuo:

$$P(X_1 \leq x_1 \mid X_2 \leq x_2, ..., X_n \leq x_n) = \frac{P(X_1 \leq x_1, X_2 \leq x_2, ..., X_n \leq x_n)}{P(X_2 \leq x_2, ..., X_n \leq x_n)} \tag{21}$$

1.6.3.3 Outras relações importantes

- O valor esperado de uma soma de variáveis aleatórias é igual à soma dos valores esperados de cada variável aleatória. Se $c_1, ..., c_n$ são constantes arbitrárias, então:

$$E[\sum_{i=1}^{n} c_i X_i] = \sum_{i=1}^{n} c_i E[X_i] \tag{22}$$

- Se as variáveis aleatórias $X_1, X_2, ..., X_n$ são independentes, então o valor esperado do produto entre as variáveis aleatórias é igual ao produto entre os valores esperados de cada variável aleatória:

$$E[\prod_{i=1}^{n} X_i] = \prod_{i=1}^{n} E[X_i] \tag{23}$$

- A covariância de duas variáveis aleatórias X e Y é um meio de medir o grau de dependência entre X e Y, e é definida por:

$$cov(X, Y) = E[(X - E[X])(Y - E[Y])] \tag{24}$$

2 Cadeias de Markov

2.1 Processo de Markov

Um processo estocástico descreve o comportamento de um conjunto de variáveis aleatórias em relação a um parâmetro de interesse, que geralmente é o tempo. Ele é definido como uma família de variáveis aleatórias X_t : $t \in T$ onde cada variável X_t é indexada pelo parâmetro $t \in T$, chamado de parâmetro de tempo caso $T \subseteq R_+ = [0, \infty)$. O conjunto de todos os possíveis valores de X_t (para cada $t \in T$) é chamado de espaço de estados S do processo estocástico.

Se T é um conjunto contável e discreto, o processo estocástico é chamado de processo de parâmetro discreto e T é representado pelo conjunto N = {0,1,...}. Caso contrário é chamado de processo de parâmetro contínuo. O espaço de estados também pode ser classificado como discreto ou contínuo. Nesse estudo o foco é voltado para os processos estocásticos que têm espaço de estados discreto e espaço de parâmetros discreto, e nesse caso é comum eles serem referenciados como cadeias.[2]

Os processos de Markov são um tipo especial de processo estocástico. Eles são flexíveis e eficientes na descrição e análise de propriedades dinâmicas de um sistema.

Processos estocásticos de parâmetro contínuo podem ser caracterizados pela função de distribuição conjunta $F_X(s;t)$ de um dado conjunto de variáveis aleatórias $\{X_{t1}, X_{t2}, ..., X_{tn}\}$, sendo o vetor de parâmetros $t = (t_1, t_2, ..., t_n) \in R^n$, e o vetor de estados $s = (s_1, s_2, ..., s_n) \in R^n$, onde $t_1 < t_2 < \cdots < t_n$:

$$F_X(s;t) = P(X_{t1} \leq s_1, X_{t2} \leq s_2, ..., X_{tn} \leq s_n) \tag{25}$$

Um processo estocástico $\{X_t : t \in T\}$ é considerado um processo de

Markov se para $t_0 < t_1 < \cdots < t_n < t_{n+1}$ e para todo $s_i \in S$ a função de distribuição acumulada condicional de X_{tn+1} só depende do valor de X_{tn} e não dos valores de $X_{t0}, X_{t1}, ..., X_{tn-1}$:

$$
\begin{aligned}
& P(X_{tn+1} \leq s_{n+1} \quad | \quad X_{tn} = s_n, X_{tn-1} = s_{n-1}, ..., X_{t0} = s_0) \\
= & P(X_{tn+1} \leq s_{n+1} \quad | \quad X_{tn} = s_n)
\end{aligned} \tag{26}
$$

2.2 Cadeias de Markov

Os processos de Markov que possuem espaço de estados e espaço de parâmetros discretos são chamados de cadeias de Markov. A Equação (26) descreve a característica mais importante das cadeias de Markov, a propriedade de Markov. Informalmente essa equação pode ser interpretada da seguinte maneira: toda a história de uma cadeia de Markov é resumida pelo estado presente X_{tn} e pelo estado imediatamente anterior. Dado o estado presente, o estado futuro é condicionalmente independente dos estados passados. Essa propriedade é chamada de **memoryless**.[2],[3]

2.2.1 Cadeias de Markov de tempo discreto - DTMC

As cadeias de Markov de tempo discreto são restritas a um parâmetro discreto de tempo T, e $T \subseteq N = \{0, 1, 2...\}$. A função de probabilidade condicional que reflete a propriedade memoryless das cadeias de Markov de tempo discreto, correspondente à Equação (26), é descrita a seguir:

Um dado processo estocástico $\{X_0, X_1, ..., X_{n+1}, ...\}$ nos pontos de observação $0, 1, ...n + 1$ constitui uma DTMC se a seguinte relação, isto é, a propriedade de Markov, é verdade para todo $n \in N$ e todo $s_i \in S$:

$$
\begin{aligned}
&P(X_{n+1} = s_{n+1} \quad | \quad X_n = s_n, X_{n-1} = s_{n-1}, ..., X_0 = s_0) \\
&= P(X_{n+1} = s_{n+1} \quad | \quad X_n = s_n)
\end{aligned} \tag{27}
$$

Dado um estado inicial s_0, a cadeia de Markov evolui com o tempo, passo a passo, mudando de estado a cada ponto de observação, de acordo com as probabilidades de transição. O lado direito da equação (27) mostra a função de probabilidade condicional das transições de um estado s_n, no passo n, para o estado s_{n+1}, no passo $(n+1)$. Considerando $S = \{0, 1, 2, ...\}$, podemos escrever a função de probabilidade condicional da transição de um passo do processo do estado i para o estado j no instante de tempo n como:

$$p_{ij} = P(X_{n+1} = s_{n+1} = j \mid X_n = s_n = i) \tag{28}$$

Dado que a cadeia está em um estado i, é certo que a mesma vai para algum estado j no próximo instante de tempo, inclusive podendo ficar no mesmo estado $(i = j)$. Então pode-se concluir que $\sum_j p_{ij} = 1$, onde $0 \leq p_{ij} \leq 1$, ou seja, o somatório das probabilidades de transição de um estado i para todos os outros estados da cadeia de Markov é sempre igual a 1. As probabilidades de transição p_{ij} da cadeia de Markov são representadas por uma matriz $\mathbf{P}$ de transição, onde o somatório de todos valores pertencentes à uma mesma linha é sempre igual a 1:

$$\mathbf{P} = [p_{ij}] \begin{pmatrix} p_{00} & p_{01} & p_{0n} \\ p_{10} & p_{11} & p_{1n} \\ p_{n0} & p_{n1} & p_{nn} \end{pmatrix} \tag{29}$$

Graficamente uma cadeia de Markov de tempo discreto é representada por um diagrama de estados e transições, que é um grafo finito direcionado, onde os estados são representados pelos nós e as transições entre os estados são representadas por setas valoradas. A seguir é mostrado um exemplo:[2]

Dada a matriz $\mathbf{P}$ que representa as probabilidades de transição de um passo de uma cadeia de Markov de dois estados:

$$\mathbf{P} = \begin{pmatrix} 0.75 & 0.25 \\ 0.5 & 0.5 \end{pmatrix} \tag{30}$$

Podemos ver pela matriz que quando o estado presente é o estado 0, há a probabilidade de 0.75 de que no próximo passo o estado seja o mesmo e há a probabilidade de 0.25 de que no próximo passo o estado seja o estado 1. Quando o estado presente é o estado 1, há a probabilidade de 0.5 tanto para o próximo estado ser 0 quanto para ser 1.

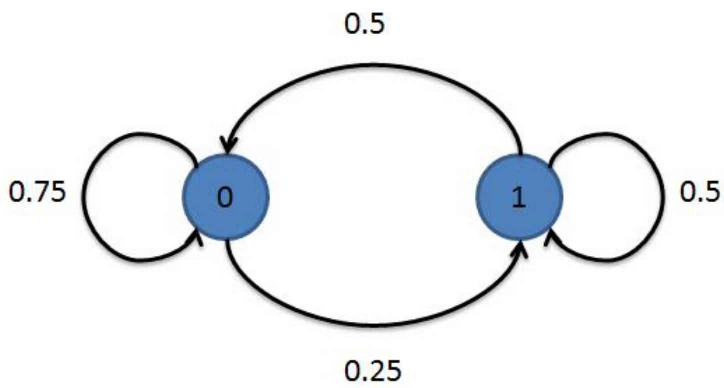


Figura 2.1 Representação gráfica da cadeia referente à matriz (30)

Como as cadeias de Markov têm a propriedade memoryless, podemos notar uma característica importante. Através da matriz de probabilidades de transição para um passo, ou seja, uma passagem de instante de tempo, podemos inferir a matriz de probabilidades de transição para n passos. Sabendo o estado atual da cadeia, podemos calcular a probabilidade de a cadeia estar em um determinado estado depois de n passos. Para fazer isso basta multiplicar a matriz $\mathbf{P}$ de probabilidades de transição de um passo por ela mesma n vezes:

$$\mathbf{P}^{(1)} = \mathbf{P} \tag{31}$$

$$\mathbf{P}^{(n)} = \mathbf{P}^{n} \tag{32}$$

No exemplo da matriz (30), para calcular a matriz de probabilidades de transição que indica a transição de um estado i para um estado j após a realização de 3 passos, se calcula:

$$\mathbf{P}^{(3)} = \mathbf{P}^{3} = \mathbf{PPP} \tag{33}$$

A maior utilidade da cadeia de Markov é a possibilidade de calcular a função de probabilidade da variável aleatória X_n, isto é, as probabilidades $v_i(n) = P(X_n = i)$ de que a cadeia de Markov esteja no estado i no instante de tempo t, após n passos. Essas probabilidades são chamadas de probabilidades de estado transiente, e permitem as medidas de performance do sistema.[2],[8]

Dada a matriz de probabilidades de transição para n passos $\mathbf{P}^{(n)}$, o vetor de probabilidades de estado $v_i(n) = (P_0(n), P_1(n), P_2(n), ...)$, que indica a probabilidade do estado i ser o estado presente da cadeia após n passos, pode ser obtido multiplicando-se o vetor inicial de probabilidades de estado, $v_i(0) = (P_0(0), P_1(0), P_2(0), ...)$ pela matriz $\mathbf{P}^{(n)}$:

$$v_i(n) = v_i(0)\mathbf{P}^{n} = v_i(n-1)\mathbf{P} \tag{34}$$

Por exemplo, assumindo que a cadeia representada pela matriz (30) sempre inicia no estado 1, então o vetor inicial de probabilidades de estado é $v_i(0) = (0, 1)$. Para encontrar o vetor de probabilidades de estado $v_i(n)$, com n $=$ 3, ou seja, após 3 passos, faz-se:

$$v_i(3) = v_i(0)\mathbf{P}^{3} = (0,1)\begin{pmatrix} 0.67 & 0.33 \\ 0.65 & 0.35 \end{pmatrix} = (0.65, 0.35) \tag{35}$$

Um vetor de probabilidades de estado de uma cadeia de Markov de tempo discreto é chamado de estacionário se as transições descritas pela matriz $\mathbf{P}$ não modificam essas probabilidades de estado, ou seja, $v_j = \sum v_i p_{ij}$.[2]

Para uma análise eficiente é interessante limitar as probabilidades de estado em um tipo particular de probabilidades de estado estacionárias, que são definidas por:

$$\tilde{v} = \lim_{n\to\infty} v_i(n) = \lim_{n\to\infty} v_i(0)\mathbf{P}^n = v_i(0) \lim_{n\to\infty} \mathbf{P}^n = v_i(0)\tilde{\mathbf{P}} \tag{36}$$

Com $n \to \infty$, é possível que a matriz $\mathbf{P}^n$ e o vetor de probabilidades de estado $v_i(n)$ tendam a convergir para $\tilde{\mathbf{P}}$ e $\tilde{v}$, respectivamente, independentemente do vetor inicial de probabilidades de estado $v_i(0)$. Se além disso $\tilde{v} > 0$ e $\sum \tilde{v} = 1$, denomina-se o vetor de probabilidades de **vetor único de estado estável** da cadeia de Markov.

Como a matriz de probabilidades de transição (30) é uma matriz pequena, fazendo a multiplicação dela por ela mesma algumas vezes podemos aproximar a matriz resultante quando esse número de multiplicações tende ao infinito, ou seja, quando $n \to \infty$. Feito isso, nota-se que a matriz (30) converge para a matriz mostrada abaixo:

$$\tilde{\mathbf{P}} = \lim_{n\to\infty} \mathbf{P}^n = \lim_{n\to\infty} \begin{pmatrix} 0.75 & 0.25 \\ 0.5 & 0.5 \end{pmatrix}^n = \begin{pmatrix} 0.67 & 0.33 \\ 0.67 & 0.33 \end{pmatrix} \tag{37}$$

A partir desse resultado podemos computar $\tilde{v}$ através da equação (36):

$$\tilde{v} = v_i(0)\tilde{\mathbf{P}} = (0.67, 0.33) \tag{38}$$

Esse vetor corresponde ao **vetor único de estado estável** da cadeia de Markov. É importante observar que como o vetor $\tilde{v}$ é independente de $v_i(0)$, o limite $\mathbf{P}^n$, quando $n \to \infty$, é independente do número de passos n e do índice i, portanto todas as linhas de $\tilde{\mathbf{P}}$ são idênticas, como acontece no caso acima, da matriz (37).[2]

O vetor único de estado estável é muito útil para extrair características de um modelo baseado em cadeias de Markov, mas ele não existe para todas as cadeias. Além disso, nem sempre é apropriado usá-lo para avaliar o modelo, dependendo dos objetivos da análise. Em alguns casos as análises têm o objetivo de analisar o comportamento do sistema a curto-prazo, ou seja, até um determinado instante de tempo no qual ele ainda sofre influência do estado inicial, portanto nesse caso o vetor único de estado estável não é útil, pois ele mostra o comportamento do sistema a longo-prazo.[2],[4]

Classificações das cadeias de Markov de tempo discreto

- **Alcançabilidade**

  Um estado j é dito ser **alcançável** por outro estado i, onde $i, j \in S$, se é possível ir do estado i para o estado j em um número finito de passos de acordo com a matriz de probabilidades de transição da cadeia de Markov. Então para algum $n \geq 1$, tem-se:[2]

  $$p_{ij}^{(n)} > 0 \tag{39}$$

  Na figura abaixo, o estado 2 é alcançável pelo estado 0, por exemplo, e o estado 0 não é alcançável pelo estado 3.

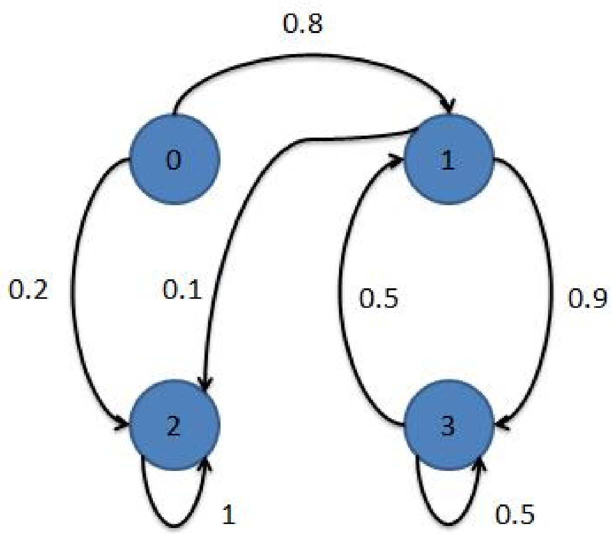


Figura 2.2 Exemplo de cadeia de Markov

- **Irredutibilidade**

  Uma cadeia de Markov de tempo discreto é chamada **irredutível** se para todo estado j, qualquer estado i pode alcançar j, ou seja, todos os estados podem alcançar todos os estados.

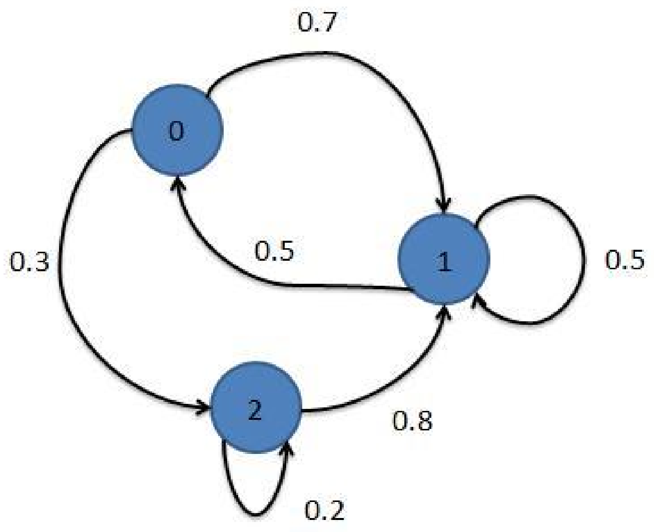


Figura 2.3 Cadeia de Markov irredutível

- **Estado absorvente**

  Um estado i é dito ser um **estado absorvente** se ele não pode alcançar nenhum estado j da cadeia:[2]

$$p_{ii} = 1 \tag{40}$$

  Na cadeia de Markov da figura 2.2 nota-se que o estado 2 é um estado absorvente.

- **Probabilidade de recorrência**

  Seja $f_i^{(n)}$ a probabilidade de recorrência para n passos, ou seja, a probabilidade de acontecer o primeiro retorno para um estado i, n passos após a cadeia sair do estado i, então a probabilidade f_i de a cadeia retornar em algum momento futuro para o estado i após sair dele é:[2]

$$f_i = \sum_{n=1}^{\infty} f_i^{(n)} \tag{41}$$

Um estado i em que $f_i = 1$ é chamado de **estado recorrente** e um estado em que $f_i < 1$ é chamado de **estado transiente**.

- **Tempo médio de recorrência**

  Dado um estado recorrente i, o tempo médio de recorrência m_i desse estado i da cadeia de Markov é:[2]

  $$m_i = \sum_{n=1}^{\infty} n f_i^{(n)} \tag{42}$$

  Se m_i é finito, i é chamado de estado recorrente não-nulo. Se $m_i = \infty$, i é chamado de recorrente nulo. Para todo estado recorrente i, seja d_i o período do estado i, então d_i é o maior divisor comum do conjunto de inteiros positivos n tal que $p_{ii}^{(n)} > 0$. Um estado recorrente é chamado de **aperiódico** se o período $d_i = 1$, e periódico se $d_i > 1$.

  Existe a prova matemática de que os estados de uma cadeia de Markov de tempo discreto irredutível são todos do mesmo tipo. Assim, se um estado de uma cadeia irredutível for aperiódico, todos os outros estados também são, e alternativamente, se um estado for periódico, todos os outros são periódicos. Uma cadeia irredutível então pode ser classificada como aperiódica ou periódica.[2]

- **Cadeia de Markov ergódica**

  Uma cadeia de Markov de tempo discreto irredutível, aperiódica, com todos os estados sendo recorrentes não-nulos e com tempo médio de recorrência finito é chamada de **ergódica**.[2]

Definições

- Os estados de uma cadeia de Markov finita e irredutível são todos recorrentes não-nulos.

- Para qualquer cadeia de Markov de tempo discreto irredutível e aperiódica, o vetor limite $\tilde{v}$ existe e é independente do vetor de probabilidades de estado inicial $v_i(0)$.

- Para uma cadeia de Markov de tempo discreto ergódica, o vetor limite $\tilde{v}$ existe e consiste no vetor único de estado estável.

2.2.2 Cadeias de Markov de tempo contínuo - CTMC

Cadeias de Markov de tempo contínuo fornecem um paradigma diferente para modelagem de sistemas. Nas CTMC's as transições entre estados podem ocorrer a qualquer momento, ao contrário das cadeias de Markov de tempo discreto, em que as transições ocorrem sempre em instantes de tempo discretos e conhecidos.[2],[3]

Um dado processo estocástico $\{X_t : t \in T\}$ constitui uma cadeia de Markov de tempo contínuo para um t_i arbitrário, com $t_i \in R^+$, $t_0 < t_1 < \cdots < t_n < t_{n+1}$, $\forall n \in N, \forall s_i \in S$, se para a função de probabilidade condicional, a seguinte relação é verdade:[2]

$$\begin{aligned} P(X_{tn+1} = s_{n+1} \quad &| \quad X_{tn} = s_n, X_{tn-1} = s_{n-1}, ..., X_{t0} = s_0) \\ = P(X_{tn+1} = s_{n+1} \quad &| \quad X_{tn} = s_n) \end{aligned} \tag{43}$$

Essa equação mostra a propriedade de Markov das cadeias de Markov de tempo contínuo. O lado direito da equação representa a probabilidade de transição $p_{ij}(u, v)$ da cadeia passar do estado i para o estado j no período de tempo que vai de u até v, com $u, v \in T$ e $u \leq v$.[2]

$$p_{ij} = P(X_v = j \mid X_u = i) \tag{44}$$

Com $\mathbf{P}(u, v) = [p_{ij}(u, v)]$ sendo a matriz de probabilidades de transição de qualquer estado i para qualquer estado j, e (u, v) um intervalo de tempo de u até v, e o vetor de probabilidades de estado $\pi_i(u) = (P_0(u), P_1(u), P_2(u), ...)$, ou seja, o vetor que indica a probabilidade de a cadeia estar no estado i no instante de tempo u, tem-se que:

$$\pi_i(v) = \pi_i(u)\mathbf{P}(u, v) \tag{45}$$

Com $u = 0$ e $v = t$, tem-se:

$$\pi_i(t) = \pi_i(0)\mathbf{P}(0, t) \tag{46}$$

Nas cadeias de Markov de tempo contínuo, as transições entre os estados podem ocorrer em qualquer instante de tempo. Para facilitar o trabalho com as CTMC, criou-se a notação de **taxa de transição**, ao invés de probabilidade de transição, como nas cadeias de Markov de tempo discreto. A taxa de transição é a taxa que indica a mudança da cadeia de um estado i para um estado j durante a passagem do tempo.[2],[3]

As taxas de transição estão relacionadas com as probabilidades condicionais de transição. A demonstração matemática da dedução que explica a notação de taxas de transição está fora do escopo desse trabalho.

As taxas de transição instantâneas são representadas por $q_{ij}(t)$, com $i \neq j$. A matriz formada pelas taxas de transição é chamada de **gerador infinitesimal $\mathbf{Q}$** da matriz de probabilidades de transição $\mathbf{P}(t) = [p_{ij}(0, t)] = [p_{ij}(t)]$.

$$\mathbf{Q} = [q_{ij}] \tag{47}$$

Como nas cadeias de Markov de tempo discreto, o vetor único de estado estável também é muito importante para as cadeias de Markov de tempo contínuo. A equação para obter as probabilidades de estado estável é:[2],[3]

$$0 = \pi\mathbf{Q} \tag{48}$$

Classificações das cadeias de Markov de tempo contínuo

- **Irredutibilidade**

  A definição de irredutibilidade nas cadeias de tempo contínuo é análoga à definição de irredutibilidade nas cadeias de tempo discreto. Uma CTMC é chamada **irredutível** se para todo estado j, qualquer estado i pode alcançar j.

- **CTMC ergódica**

  Uma cadeia de Markov de tempo contínuo irredutível é ergódica se e somente se o vetor único de estado estável π existe.

Definições

- O vetor de estado estável T, se existir, pode ser unicamente determinado pela solução do sistema linear da Equação (48).

- O vetor único de estado estável T existe se a cadeia de Markov de tempo contínuo irredutível é finita.

3 Modelagem e análise de desempenho

3.1 Medidas de desempenho

Abaixo será descrito um exemplo adaptado do livro "Availability and Reliability modeling for computer systems", de D.I.Heimann, N.Mittal, e K.S. Trivedi.[4] O objetivo desse exemplo é mostrar a modelagem de um sistema simples utilizando valores literais, para descrever algumas medidas de desempenho que podem ser retiradas do modelo, em categorias como disponibilidade, confiabilidade e performance. O exemplo utiliza uma cadeia de Markov de tempo contínuo.

O sistema corresponde a 2 processadores que estão processando alguma tarefa. Cada processador está sujeito a falhas, com uma taxa de ocorrência de falhas de γ. Em alguns tipos de falhas, o sistema consegue entrar em um estado de recuperação, essas falhas têm taxa de ocorrência $c\gamma$, onde c é a probabilidade de recuperação, com valor geralmente próximo de 1. Alternativamente, existem falhas mais graves, em que o sistema não consegue entrar em estado de recuperação e precisa ser resetado, entrando no estado de reset. Essas falhas têm taxa de ocorrência $(1-c)\gamma$. Quando os processadores falham, precisam ser reparados. A taxa de reparação é δ. Somente um processador pode ser reparado por vez. Se os dois processadores falharem, o sistema para de funcionar como um todo, não podendo entrar em estado de reparação ou de reset. Nesse caso o sistema só volta a funcionar quando um dos processadores é reparado. A taxa de saída do estado de reparação é β, e a taxa de saída do estado de reset é α.

O espaço de estados da CTMC é $S = \{2, RC, RS, 1, 0\}$. Os estados 2, 1 e 0 indicam a quantidade de processadores em funcionamento normal. RC é o estado de recuperação do sistema, e RS é o estado de reset do sistema.

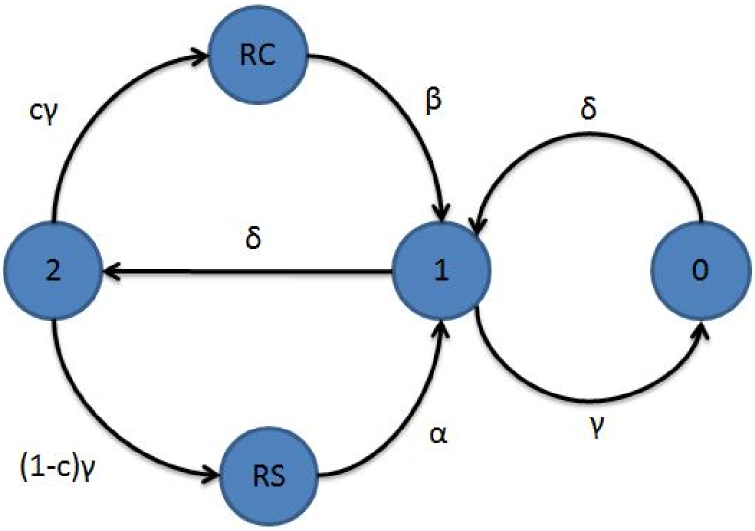


Figura 3.1 Diagrama de estados e transições

O gerador infinitesimal da CTMC é $\mathbf{Q}$:

$$\mathbf{Q} = \begin{pmatrix} -\gamma & c\gamma & (1-c)\gamma & 0 & 0 \\ 0 & -\beta & 0 & \beta & 0 \\ 0 & 0 & -\alpha & \alpha & 0 \\ \delta & 0 & 0 & -(\delta+\gamma) & \gamma \\ 0 & 0 & 0 & \delta & -\delta \end{pmatrix} \tag{49}$$

Esse modelo é uma abstração do sistema, mas dependendo de quais características são relevantes para a análise do sistema dentro de um contexto, ele pode ser modificado para atender a um certo tipo de análise. Abaixo serão descritos exemplos de tipos de análises que podem ser feitas e como o modelo é alterado para cada tipo, caso isso seja útil.

- **Disponibilidade** - A disponibilidade de um sistema indica o grau de atendimento do sistema quando o mesmo é requisitado. Um sistema muito disponível quase sempre está pronto para executar um serviço, enquanto um sistema pouco disponível passa muito tempo "fora do ar". Geralmente, pequenos intervalos de tempo de indisponibilidade são aceitáveis, mas o nível de disponibilidade mínimo varia de sistema para sistema.[2],[4] Analisando o exemplo da Figura 3.1, primeiramente os modeladores têm que definir o tempo máximo tolerável de indisponibilidade do sistema, ou seja, quanto tempo o sistema pode ficar indisponível. Depois os estados da cadeia devem ser classificados como "up"ou "down". Os estados "up"são os estados em que o sistema está funcionando corretamente, e os estados "down"são os estados em que o sistema está "fora do ar"por algum motivo.

No exemplo da Figura 3.1, os estados "down"são {RC,RS,0}. Alternativamente, se o tempo de recuperação do sistema for muito pequeno, por exemplo, o estado RC pode até ser retirado do modelo, dependendo do contexto.

Finalmente, o modelador tem que decidir quais medidas retirar do modelo, dependendo do seu objetivo. São exemplos de medidas de disponibilidade: a probabilidade do sistema estar em um estado "up"em um determinado tempo t, dado um estado inicial, e a proporção de tempo que a cadeia passa em um determinado estado "up"no período de 0 a t.

- **Confiabilidade** - A confiabilidade define o grau de interruptibilidade de um sistema. Um sistema muito confiável dificilmente sofre interrupções durante a execução de um serviço, enquanto que um sistema pouco confiável está sujeito a muitas interrupções durante a execução de um serviço. O modelador precisa definir que estados são considerados interrupções de serviço, e se essas interrupções são toleradas ou não pelo sistema. Se for definido que o estado de reset e o estado 0 são interrupções intoleráveis, por exemplo, o modelador pode modificar o modelo, transformando esses estados em estados absorventes, como pode-se ver na figura 3.2.[2],[4]

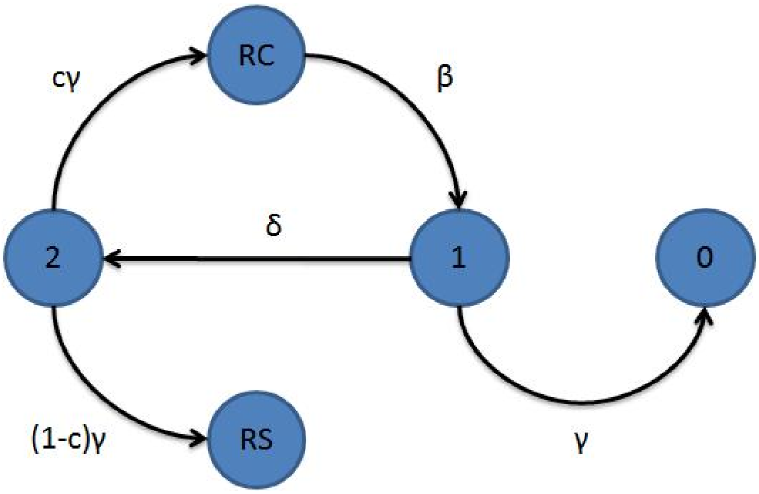


Figura 3.2 Exemplo com estados absorventes capturando requisitos de confiabilidade

- **Performance** - A performance do sistema define o quão bem o sistema executa suas tarefas. As medidas geralmente informam como o sistema utiliza seus recursos e a velocidade de processamento do mesmo. Um modo de medir a performance do sistema é selecionar quais estados representam a maior eficiência.

  No exemplo da Figura 3.1, nota-se que o sistema está em capacidade máxima no estado 2, no qual os dois processadores estão em funcionamento, portanto é interessante medir se o sistema permanece por muito tempo nesse estado de capacidade máxima. Em casos de sistemas complexos, deve-se definir índices de capacidade para cada estado afim de comparar as diferentes configurações possíveis.[2],[5]

3.2 Soluções analíticas

As soluções analíticas se baseiam em derivar equações do modelo construído para obter informações do comportamento dinâmico do sistema. Isso pode ser feito se a cadeia apresenta uma estrutura regular. No exemplo de solução analítica apresentado neste trabalho a cadeia gerada possui uma estrutura que permite a representação do sistema por equações simples. Em sistemas complexos, nos quais a cadeia gerada pode não ter uma estrutura favorável, é necessário utilizar algoritmos de matemática numérica para encontrar resultados aproximados.[2]

3.2.1 Exemplo de solução analítica utilizando um modelo de DTMC

Um sistema computacional consiste de dois servidores de nomes idênticos trabalhando em paralelo. Esses servidores recebem requisições e as atendem. O tempo gasto para o servidor responder à requisição depois de processá-la é desprezado. A passagem do tempo é representada por intervalos indexados por $k = 0, 1, 2, 3, ...$ As seguintes regras definem o funcionamento do sistema:

- Somente uma requisição pode ser submetida ao sistema por intervalo de tempo. Este evento ocorre com probabilidade α.
- Quando uma requisição é submetida ao sistema ela é atendida pelo servidor disponível.
- Se ambos os servidores estão disponíveis, a requisição é atendida pelo primeiro servidor.
- Se ambos os servidores estão ocupados, a requisição não é atendida e é perdida.
- Quando um servidor está ocupado, a probabilidade de terminar o processamento da requisição em cada intervalo é β.

- Se uma requisição é submetida ao servidor em um intervalo em que os dois servidores estão ocupados e um dos servidores completa o processamento neste intervalo, então a requisição que chegou é processada.

- Se os dois servidores estão ocupados e os dois terminam o processamento no mesmo intervalo de tempo, somente um dos servidores (o primeiro) pode ser liberado nesse intervalo de tempo, assim o outro servidor só é liberado no próximo intervalo.

Percebe-se que esse sistema pode ser modelado por cadeias de Markov, já que ele pode ser representado por estados e transições entre esses estados e possui a propriedade **memoryless**.

Analisando as regras, o conjunto dos estados do sistema é S = {2SL,1SO, 2SO}, sendo 2SL o estado em que os 2 servidores estão livres, 1SO o estado em que o primeiro servidor está ocupado e o segundo está livre, e 2SO o estado em que os 2 servidores estão ocupados.

Após definir os estados, deve-se determinar as transições entre os estados a partir das regras. Isso gera o diagrama de estados e transições a seguir:

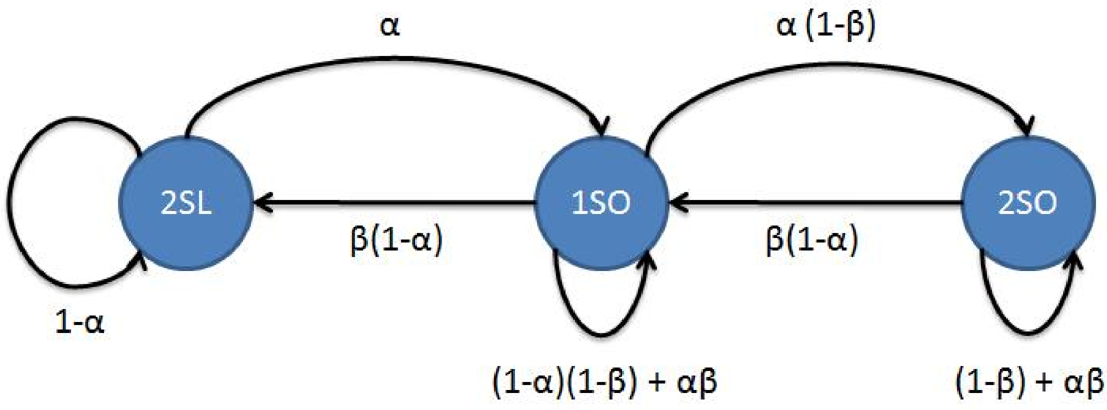


Figura 3.3 Diagrama de estados e transições da DTMC

Abaixo seguem as descrições das probabilidades que definem as transições entre os estados:

- α = Probabilidade de chegar uma requisição
- $1 - \alpha$ = Probabilidade de não chegar uma requisição
- $\beta(1 - \alpha)$ = Probabilidade de um servidor terminar de processar uma requisição e não chegar outra requisição
- $(1 - \alpha)(1 - \beta)$ = Probabilidade de não chegar uma requisição e do servidor não terminar de processar uma requisição
- $\alpha\beta$ = Probabilidade de chegar uma requisição e de o servidor termimar de processar uma requisição
- $\alpha(1 - \beta)$ = Probabilidade de chegar uma requisição e de o servidor não terminar de processar uma requisição

Os valores de α e β variam de acordo com o contexto. Se o tempo for indexado em segundos, por exemplo, a probabilidade α de chegar uma requisição é menor do que se o tempo for indexado em minutos. No caso desse exemplo cada intervalo de tempo vai ser definido como sendo de 20 segundos, portanto quando $k = 0$, $t = 0$, $k = 1$, $t = 20s$, $k = 2$, $t = 40s$, e assim por diante. As probabilidades α e β vão ser fixadas em 0.3 e 0.5, respectivamente. Aplicando esses valores o diagrama de estados e transições fica como abaixo:

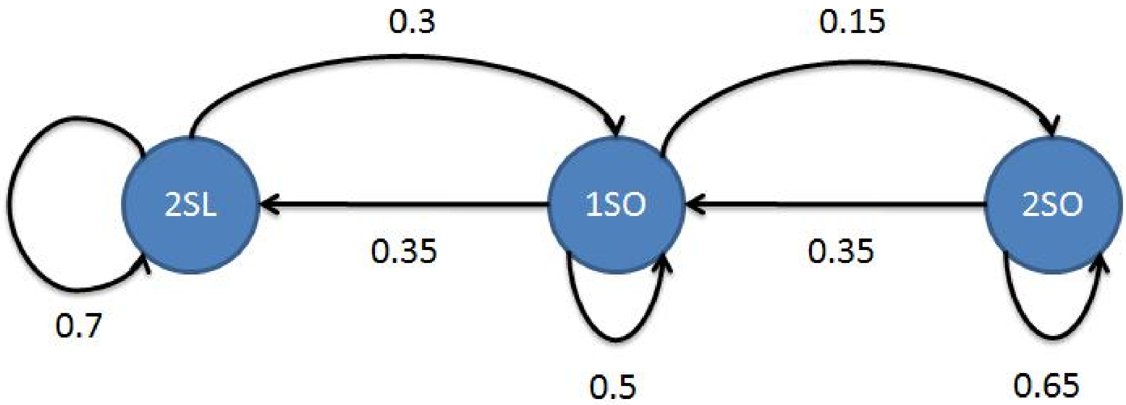


Figura 3.4 Diagrama de estados e transições da DTMC

O próximo passo é definir a matriz de transição **P**. Considerando o estado 2SL como o estado 0, o estado 1SO como estado 1 e o estado 2SO como estado 2, a matriz segue abaixo:

$$\mathbf{P} = \begin{pmatrix} 0.7 & 0.3 & 0 \\ 0.35 & 0.5 & 0.15 \\ 0 & 0.35 & 0.65 \end{pmatrix} \tag{50}$$

Agora que o modelo foi construído, pode-se avaliar suas características. A cadeia de Markov gerada é irredutível, pois para todo estado j, qualquer estado i pode alcançar j, ou seja, todos os estados podem alcançar todos os estados. A cadeia também é aperiódica, pois todos os estados são aperiódicos, ou seja, todo estado i pode continuar no estado i no próximo passo da cadeia com probabilidade maior que 0. Sendo assim, a cadeia é ergódica, pois é irredutível, aperiódica e todos os estados são recorrentes não-nulos.

Como a cadeia é ergódica, o vetor único de estado estável existe, e pode ser definido por:

$$\tilde{v} = v_i(0)\tilde{\mathbf{P}} \tag{51}$$

A matriz $\tilde{\mathbf{P}}$ é obtida encontrando-se o limite:

$$\tilde{\mathbf{P}} = \lim_{n\to\infty} \mathbf{P}^n = \lim_{n\to\infty} \begin{pmatrix} 0.7 & 0.3 & 0 \\ 0.35 & 0.5 & 0.15 \\ 0 & 0.35 & 0.65 \end{pmatrix}^n \tag{52}$$

Para encontrar uma matriz aproximada de $\tilde{\mathbf{P}}$, multiplica-se $\mathbf{P}$ algumas vezes até ela convergir. Nesse caso a matriz foi multiplicada por ela mesma 10 vezes, ou seja, fez-se $\mathbf{P}^{10}$.

Abaixo se encontram as matrizes para n passos da cadeia, com $1 \leq n \leq 10$ e com duas casas decimais de precisão nos valores. As matrizes foram representadas de forma a economizar espaço e para melhorar a análise da convergência dos valores:

- Matriz $\mathbf{P}$ = {{0.7,0.3,0},{0.35,0.5,0.15},{0,0.35,0.65}}
- Matriz $\mathbf{P}^2$ = {{0.59,0.37,0.05},{0.43,0.40,0.17},{0.13,0.40,0.47}}
- Matriz $\mathbf{P}^3$ = {{0.55,0.37,0.08},{0.44,0.39,0.17},{0.23,0.40,0.37}}
- Matriz $\mathbf{P}^4$ = {{0.51,0.37,0.12},{0.44,0.39,0.17},{0.30,0.39,0.31}}
- Matriz $\mathbf{P}^5$ = {{0.50,0.38,0.12},{0.44,0.39,0.17},{0.36,0.39,0.25}}
- Matriz $\mathbf{P}^6$ = {{0.47,0.38,0.15},{0.44,0.39,0.17},{0.38,0.39,0.23}}
- Matriz $\mathbf{P}^7$ = {{0.46,0.38,0.16},{0.44,0.39,0.17},{0.41,0.39,0.20}}
- Matriz $\mathbf{P}^8$ = {{0.45,0.38,0.17},{0.44,0.39,0.17},{0.43,0.39,0.18}}
- Matriz $\mathbf{P}^9$ = {{0.44,0.39,0.17},{0.44,0.39,0.17},{0.44,0.39,0.17}}
- Matriz $\mathbf{P}^{10}$ = {{0.44,0.39,0.17},{0.44,0.39,0.17},{0.44,0.39,0.17}}

Nota-se que aproximando os valores por duas casas decimais a matriz começa a se repetir a partir de 9 passos, essa é a quantidade média de passos para essa cadeia não ser mais influenciada pelo estado inicial. Assumindo que no estado inicial os servidores estejam livres, calcula-se o vetor único de estado estável como:

$$\tilde{v} = v_i(0)\tilde{\mathbf{P}} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} . \begin{pmatrix} 0.44 & 0.39 & 0.17 \\ 0.44 & 0.39 & 0.17 \\ 0.44 & 0.39 & 0.17 \end{pmatrix} \tag{53}$$

Assim, o vetor único de estado estável da cadeia, que indica as probabilidades de estado quando o sistema entra em equilíbrio, não sofrendo mais influência do estado inicial, é:

$$\tilde{v} = \{0.44, 0.39, 0.17\} \tag{54}$$

Para verificar se esse vetor realmente é o vetor de estado estável, basta usar a equação abaixo:

$$\tilde{v} = \tilde{v}\mathbf{P} \tag{55}$$

Calculando a multiplicação entre o vetor de estado estável e a matriz de transição $\mathbf{P}$, nota-se que o resultado é o mesmo vetor, portanto esse realmente é o vetor único de estado estável da cadeia.

$$\tilde{v} = \begin{pmatrix} 0.44 & 0.39 & 0.17 \end{pmatrix} . \begin{pmatrix} 0.7 & 0.3 & 0 \\ 0.35 & 0.5 & 0.15 \\ 0 & 0.35 & 0.65 \end{pmatrix} = \begin{pmatrix} 0.44 & 0.39 & 0.17 \end{pmatrix} \tag{56}$$

Como pode-se ver, com o sistema em estado estável, a probabilidade de o sistema estar com os dois servidores livres é de 44%, com um servidor ocupado é 39%, e com os dois servidores ocupados é de 17%.

Utilizando a matriz de transição $\mathbf{P}$, assumindo que o sistema começa no estado 2SL e lembrando que cada passo da cadeia representa 20 segundos, várias informações sobre o funcionamento do sistema podem ser inferidas, como:

- **Probabilidade do sistema estar totalmente disponível, ou seja, os dois servidores estarem livres, após 3 passos, ou seja, após 1 minuto.**

  Para isso faz-se a multiplicação do vetor inicial de probabilidades de estado com a matriz $\mathbf{P}^3$, assim obtendo o vetor de probabilidades de estado para 3 passos.

$$\begin{pmatrix} 1 & 0 & 0 \end{pmatrix} . \begin{pmatrix} 0.55 & 0.37 & 0.08 \\ 0.44 & 0.39 & 0.17 \\ 0.23 & 0.40 & 0.37 \end{pmatrix} = \begin{pmatrix} 0.55 & 0.37 & 0.08 \end{pmatrix} \tag{57}$$

  Observando a equação (57) nota-se que a probabilidade do sistema estar no estado 2SL, ou seja, totalmente disponível, é de 55%.

- **Probabilidade de pelo menos um dos servidores do sistema estar ocupado após 6 passos, ou seja, após 2 minutos.**

  Para isso faz-se a multiplicação do vetor inicial de probabilidades de estado com a matriz $\mathbf{P}^6$, assim obtendo o vetor de probabilidades de estado para 6 passos.

$$\begin{pmatrix} 1 & 0 & 0 \end{pmatrix} . \begin{pmatrix} 0.47 & 0.38 & 0.15 \\ 0.44 & 0.39 & 0.17 \\ 0.38 & 0.39 & 0.23 \end{pmatrix} = \begin{pmatrix} 0.47 & 0.38 & 0.15 \end{pmatrix} \tag{58}$$

  Para obter a probabilidade de pelo menos um dos servidores estar ocupado após 2 minutos, basta somar as probabilidades dos estados 1SO e 2SO, assim obtendo o valor de 53%.

- **Probabilidade do sistema permanecer totalmente ocupado por n passos.**

Olhando para a cadeia de Markov em questão, representada na Figura 3.4, observa-se que quando a cadeia está totalmente ocupada, ou seja, no estado 2SO, a probabilidade da mesma continuar totalmente ocupada no próximo passo da cadeia é de 65%. Sendo assim, como a cadeia possui a propriedade memoryless, se a cadeia continuar no estado 2SO no próximo passo, a probabilidade de continuar no mesmo estado vai ser novamente de 65%, pois esse evento não depende do passado da cadeia. Portanto, para calcular a probabilidade do sistema continuar no estado 2SO por n passos basta fazer $0,65^n$. Por exemplo, a probabilidade do sistema permanecer totalmente ocupado por 3 minutos, ou seja, 9 passos, é de $0.65^9 = 0.02 = 2\%$.

3.3 Soluções de simulação

Em vários modelos é difícil ou impossível encontrar soluções analíticas para extrair medidas de desempenho. Isso acontece quando a cadeia é complexa e não possui uma estrutura da qual se conhece teorias para derivação de equações, ou quando a complexidade computacional para encontrar a solução do modelo por meio de algoritmos numéricos é muito alta. Nessas situações, a alternativa é simular o funcionamento do sistema "executando"o modelo e observando o seu comportamento quanto às características desejadas. A simulação não é feita somente quando não se pode utilizar soluções analíticas, na verdade, muitas vezes a simulação é escolhida por ser um método muito flexível, mesmo existindo soluções analíticas viáveis para o modelo. [2],[3]

A solução de simulação é um experimento para determinar as características do sistema empiricamente. É um método que imita o comportamento do sistema com a passagem do tempo. Esse método consiste de modelar um sistema hipotético ou existente, realizar o experimento (executar o modelo) e analisar estatisticamente os resultados da execução.

3.3.1 Exemplo de modelagem utilizando o software Relex Markov

Abaixo será mostrado um exemplo de simulação de um modelo baseado em cadeias de Markov, utilizando o software Relex 2009. Esse software foi desenvolvido pela empresa PTC e fornece várias técnicas para avaliação de características de sistemas. Uma das subdivisões do software é a **Relex Markov**, responsável por permitir a simulação de uma cadeia de Markov, com várias opções de análise estatística do modelo. Essa subdivisão do programa foi usada para construir o exemplo abaixo, com o auxílio do manual do software. Para entrar no modo Relex Markov, após abrir o programa, clica-se no botão indicado na Figura 3.5:

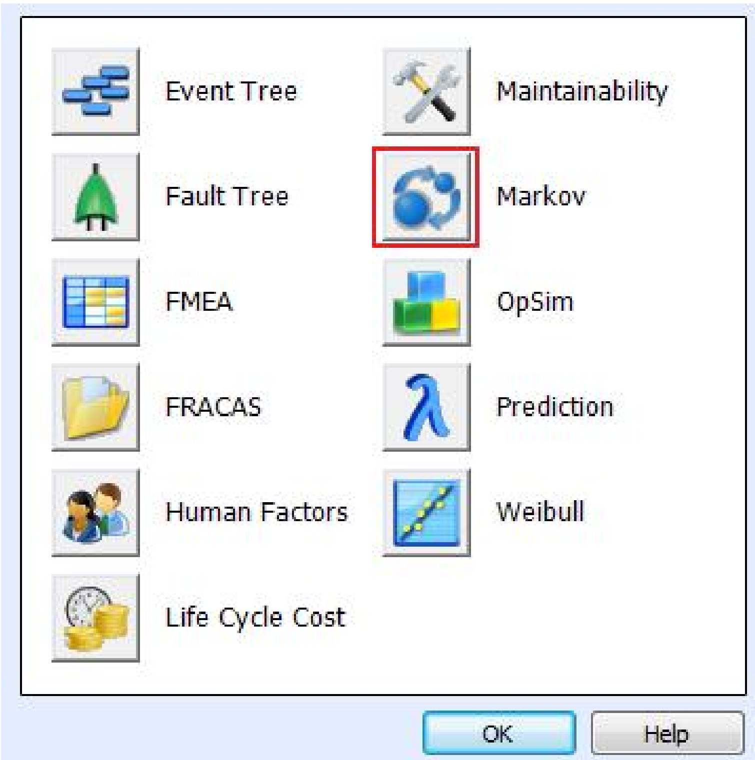


Figura 3.5 Como entrar no modo Relex Markov

Após isso, cria-se um novo projeto e clica-se no campo "**Click here to insert a new Markov Diagram**". Depois coloca-se o nome do diagrama de estados e transições a ser criado, em seguida colocando um nome para o mesmo. Neste exemplo o nome do diagrama é "Sistema de memória". Um estado inicial é automaticamente criado.

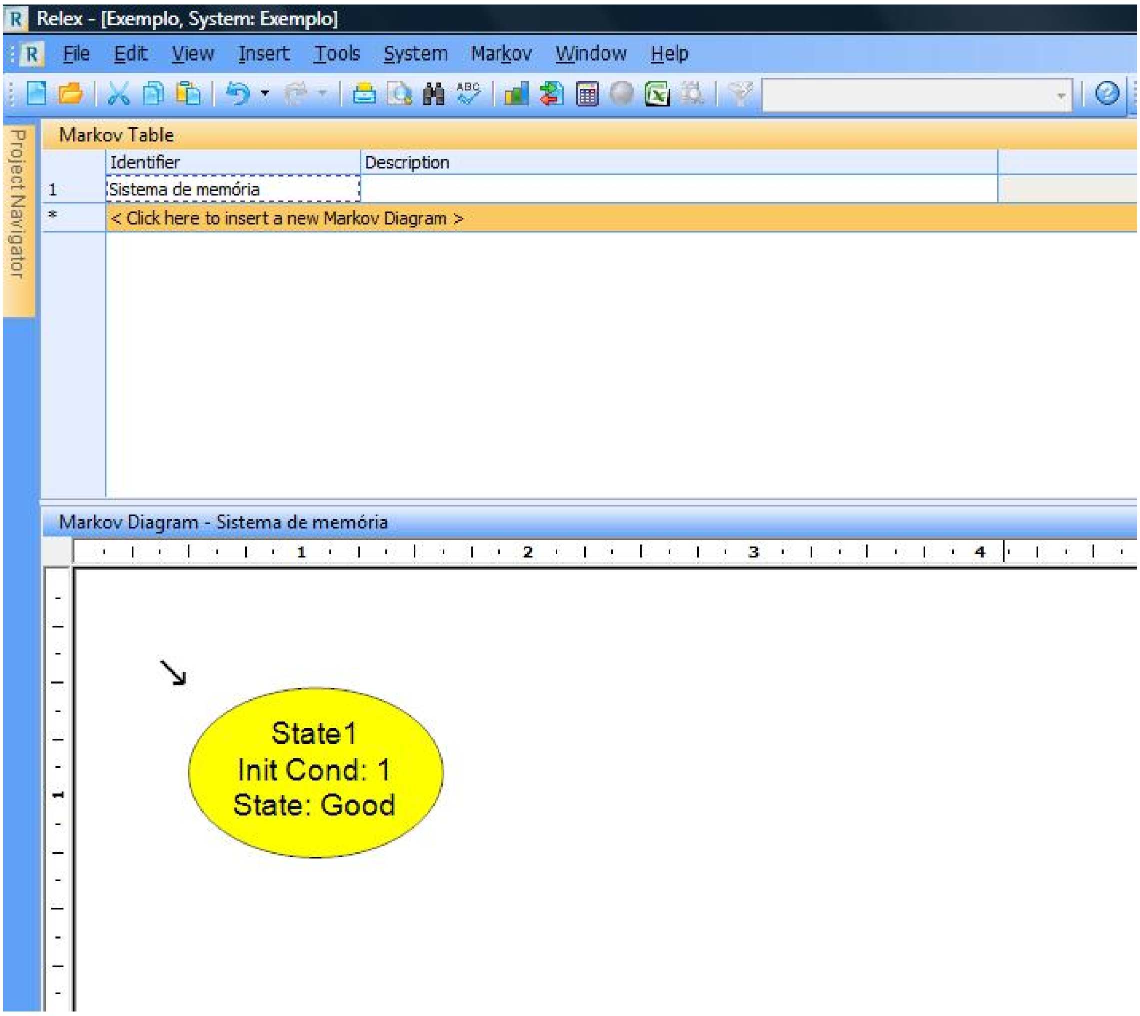


Figura 3.6 Tela inicial do modo Relex Markov

Como pode-se ver na Figura 3.6, a tela é dividida horizontalmente. A parte superior é usada para definir os nomes dos diagramas e navegar entre eles, caso exista mais de um, e a parte inferior é usada para construir e visualizar os diagramas de transição.

O estado inicial da cadeia é indicado por uma seta preta que aponta para o mesmo, pode-se ver essa seta na Figura 3.6. Para inserir um estado, clica-se na opção "**Insert** → **State**". Quando essa opção é escolhida, uma janela aparece. Essa janela é mostrada na Figura 3.7. O programa fornece diferentes formas de representar graficamente um estado. O usuário pode personalizar essas formas, mas por padrão existem 4 formas, que são: uma elipse amarela, um losango azul, um retângulo verde e um retângulo vermelho. A utilidade dessa opção vai ser explicada em breve.

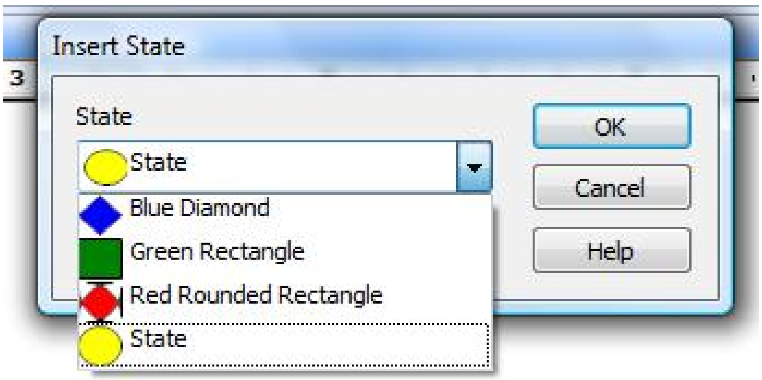


Figura 3.7 Tela de inserção de estado

Para definir um estado como sendo o estado inicial basta clicar sobre o mesmo com o botão direito do mouse e escolher a opção "**Set as Inicial State**". Ao clicar duas vezes com o botão esquerdo do mouse sobre um estado, o usuário pode definir as características do mesmo, na tela de propriedades do estado, mostrada na Figura 3.8.

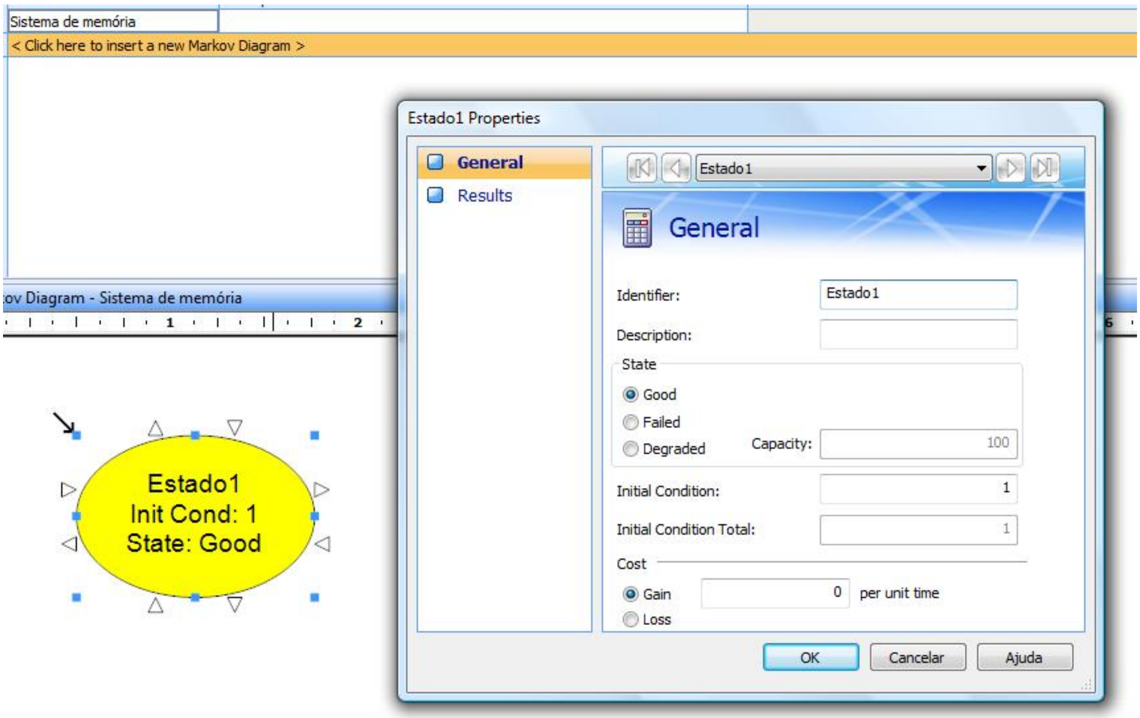


Figura 3.8 Tela de propriedades de estado

Na tela mostrada acima, o campo "**Identifier**"é usado para definir um nome para o estado. O campo "**Description**"serve para colocar alguma descrição breve sobre o estado, caso o usuário queira. Logo abaixo o usuário escolhe a condição do sistema naquele estado. O usuário pode escolher se o sistema naquele estado está funcionando normalmente, com a opção "**Good**", se o sistema está em estado de falha, com o estado "**Failed**", ou se o sistema está funcionando de forma degradada, com a opção "**Degraded**", inclusive nesse caso escolhendo o nível de degradação através do campo "**Capacity**", que indica para que nível vai a capacidade do sistema caso ele entre nesse estado.

Depois de criar os estados e definir qual é o estado inicial, é necessário criar as transições entre os estados. Para isso basta clicar com o botão esquerdo do mouse na borda de um estado e arrastar a seta que aparece para o estado alvo. Clicando duas vezes na seta pode-se definir as propriedades da transição, isso é mostrado na Figura 3.9:

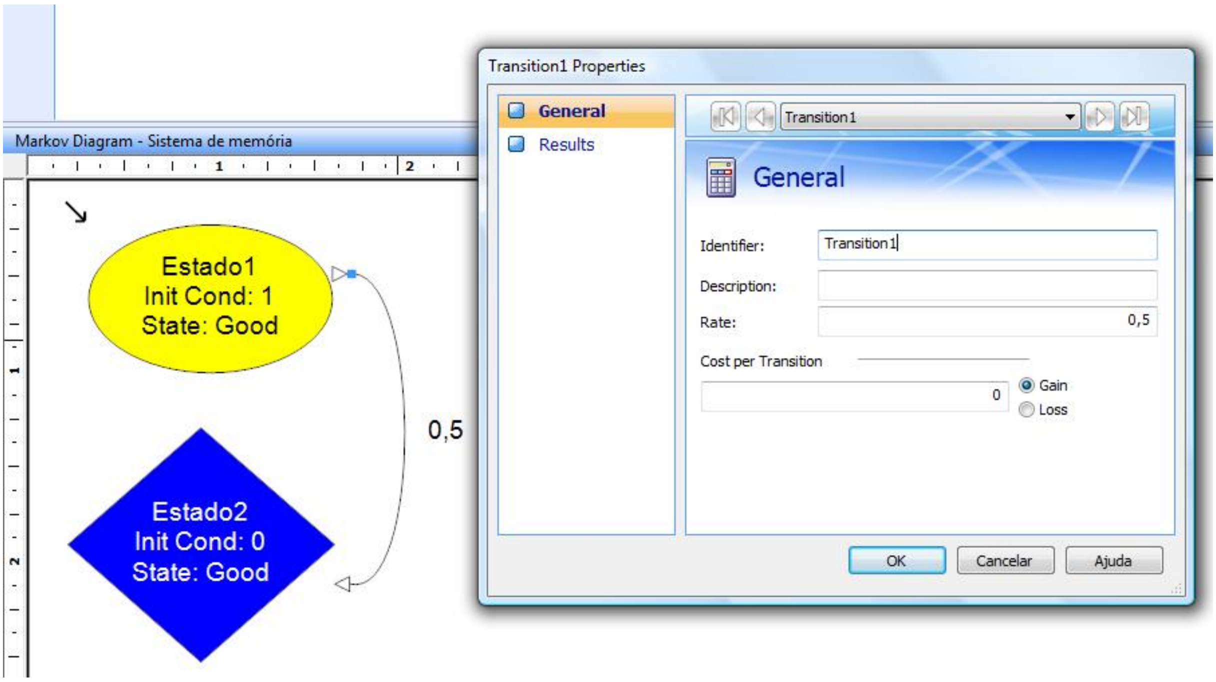


Figura 3.9 Tela de propriedades de transição

Os campos "**Identifier**"e "**Description**"da figura acima já são conhecidos. O campo "**Rate** é o campo utilizado para definir a taxa de transição entre os estados.

Agora que foi visto como é possível modelar um sistema como uma cadeia de Markov através do software Relex Markov, vai ser descrito um exemplo de avaliação de um sistema de memória simples.

O sistema de memória de um determinado computador de bolso consiste em 4 módulos de memória SRAM. Cada módulo está sujeito a falhas, e uma falha em um módulo de memória ocorre com taxa de 0,01 vezes/hora. Com o sistema em funcionamento, somente um módulo pode ser reparado por vez, e isso acontece com taxa de 0,1 vezes/hora. Quando somente um módulo de memória falha, o sistema funciona com capacidade de 75%. Quando dois módulos falham, o sistema passa a funcionar com 50% de capacidade. Quando três módulos falham, o sistema passa a funcionar com capacidade de 25%. Finalmente, quando os quatro módulos falham, o sistema para de funcionar. Se o sistema encontra-se no estado de falha total, ele pode ser completamente reparado, ou seja, todos os módulos de memória voltam a funcionar corretamente, e isso acontece com taxa de 0,1 vezes/hora. O sistema modelado pode ser visto na figura abaixo:

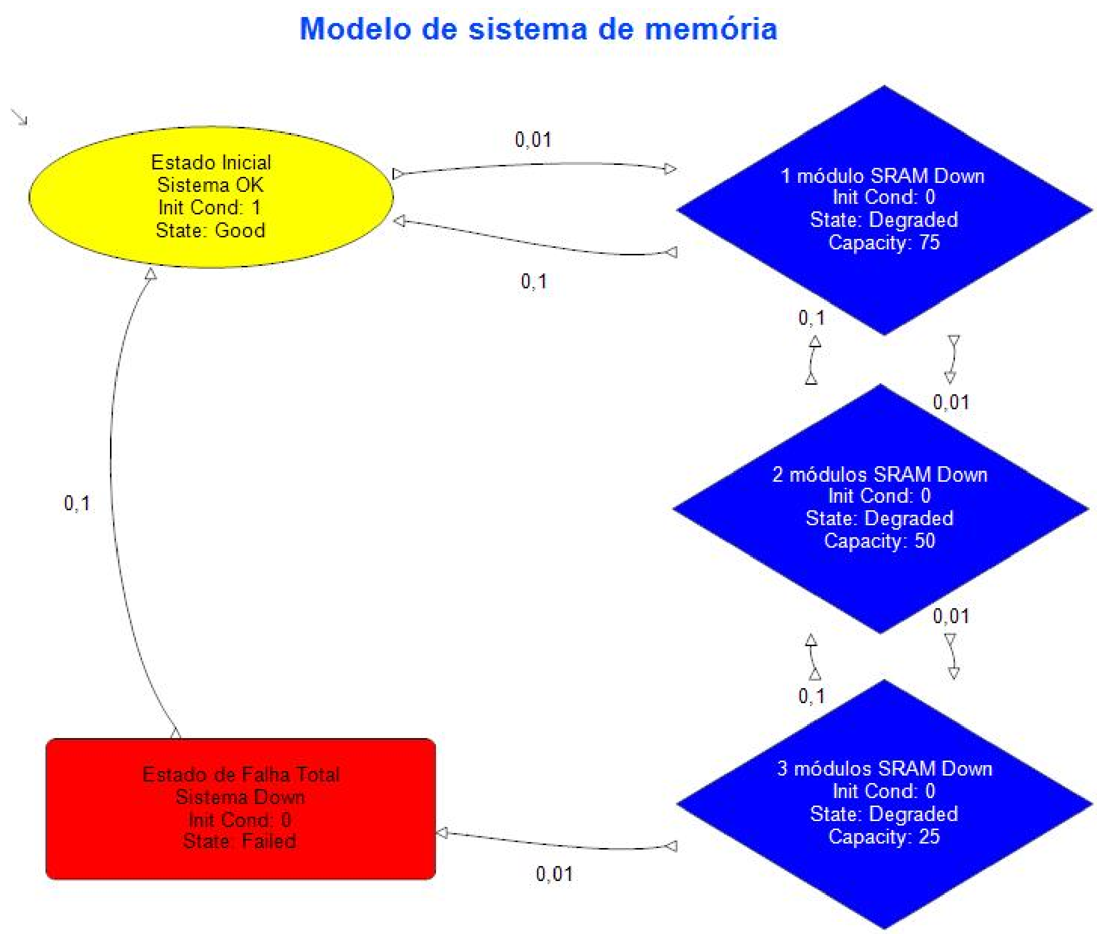


Figura 3.10 Modelo de sistema de memória modelado no software Relex Markov

Na figura acima pode-se notar a utilidade da opção de mostrar os estados com diferentes cores e formas. O estado representado pela elipse amarela é o estado de funcionamento normal do sistema (Good), os estados representados por losangos azuis são os estados em que o sistema funciona de forma degradada (Degraded), por causa das falhas nos módulos de memória, e o estado representado pelo retângulo vermelho é o estado de falha total do sistema (Failed). Diferenciar os estados com cores e formas faz com que o modelo seja mais legível e intuitivo.

Depois de criar o modelo, o usuário deve simular o funcionamento do sistema através da opção **Calculate**, que pode ser ativada pelo menu superior do programa ou apertando-se a tecla F8 do teclado. Ao ativar a opção **Calculate** o usuário precisa definir os parâmetros de simulação. Nas figuras seguintes serão mostradas as telas da opção **Calculate**. Todos os valores de tempo têm unidade **hora**.

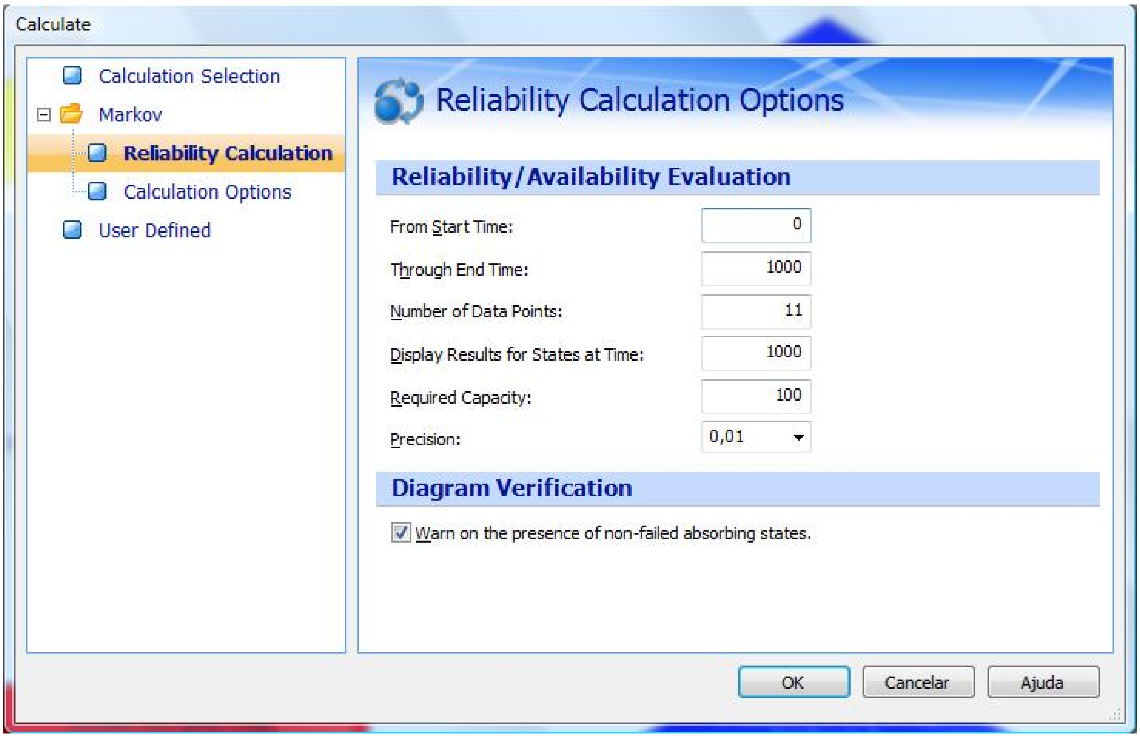


Figura 3.11 Primeira tela da opção **Calculate**

Na Figura 3.11, o campo **From Start Time** é usado para definir o instante inicial de avaliação, o campo **Through End Time** é usado para definir o instante final de avaliação, o campo **Number of Data Points** é utilizado para definir o número de avaliações parciais e o campo **Required Capacity** serve para definir qual é o valor mínimo de capacidade para o sistema funcionar perfeitamente.

No caso desse exemplo foi definido um tempo de 1000 horas de simulação de funcionamento do sistema, com avaliações parciais a cada intervalo de 100 horas. A capacidade mínima para o sistema ser considerado funcionando perfeitamente é de 100%.

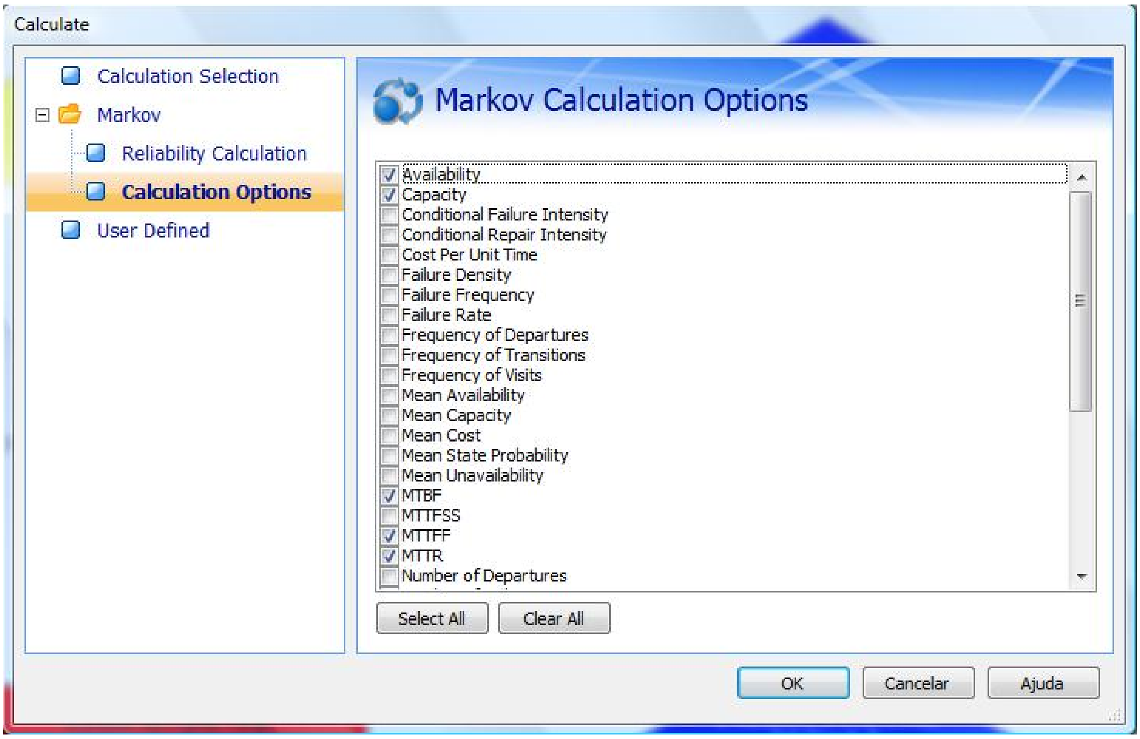
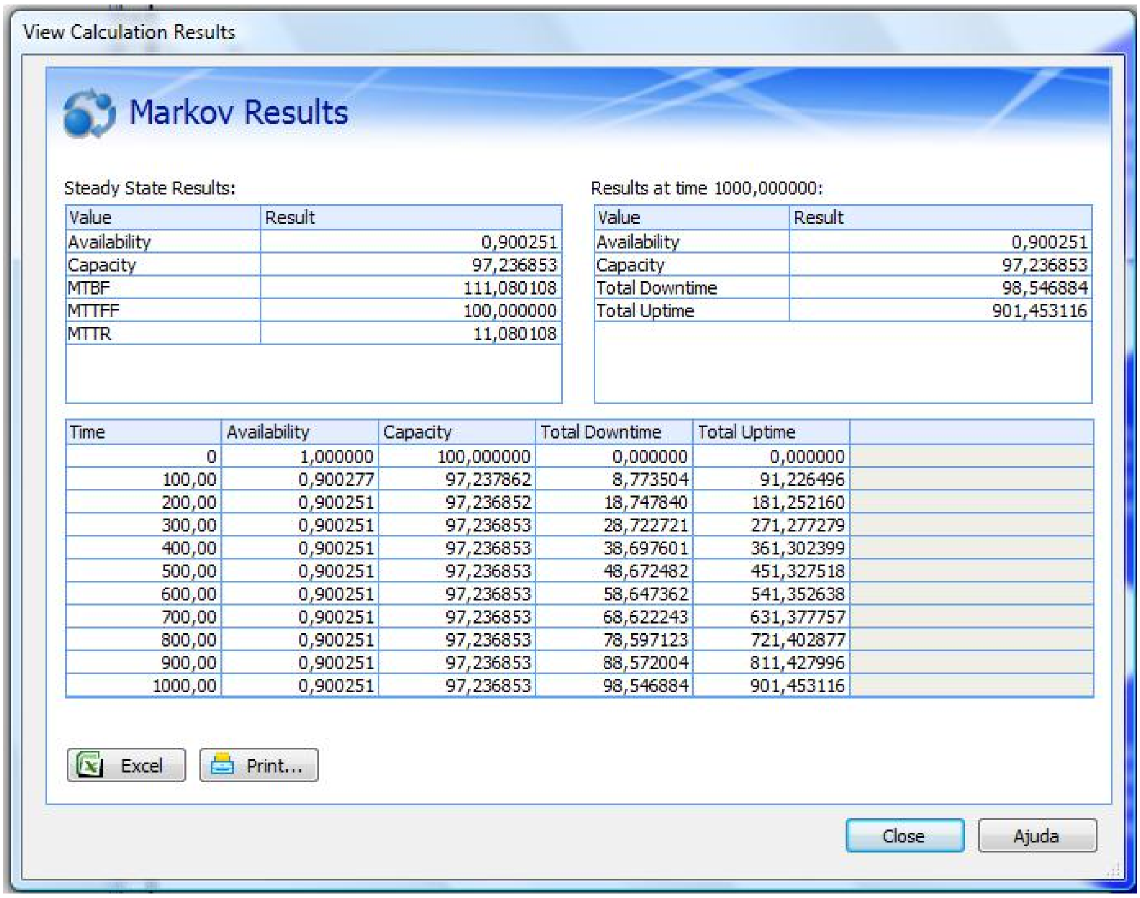


Figura 3.12 Segunda tela da opção **Calculate**

A Figura 3.12 mostra as opções de medidas de desempenho que o usuário pode escolher para que sejam calculadas durante a simulação. Existem vários tipos de medidas, de modo que para esse exemplo só serão descritas algumas delas, que são:

- **Availability** - É a probabilidade de que o sistema esteja funcionando corretamente.
- **Capacity** - É a capacidade média de funcionamento do sistema.
- **Total downtime** - É o tempo que o sistema passou no estado de falha (Down).
- **Total uptime** - É o tempo que o sistema passou no estado de funcionamento correto (Up).
- **MTBF** - É o tempo médio entre a ocorrência de falhas consecutivas. (Mean Time Between Failures).
- **MTTFF** - É o tempo médio para ocorrer a primeira falha no sistema. (Mean Time To First Failure).
- **MTTR** - É o tempo médio que o sistema leva para reparar uma falha. (Mean Time To Repair).
- **Time Spent in State** - É o tempo de permanência do sistema em determinado estado.

Na figura abaixo pode-se ver os valores calculados pelo software.



Value	Result
Availability	0,900251
Capacity	97,236853
MTBF	111,080108
MTTFF	100,000000
MTTR	11,080108

Value	Result
Availability	0,900251
Capacity	97,236853
Total Downtime	98,546884
Total Uptime	901,453116

Time	Availability	Capacity	Total Downtime	Total Uptime
0	1,000000	100,000000	0,000000	0,000000
100,00	0,900277	97,237862	8,773504	91,226496
200,00	0,900251	97,236852	18,747840	181,252160
300,00	0,900251	97,236853	28,722721	271,277279
400,00	0,900251	97,236853	38,697601	361,302399
500,00	0,900251	97,236853	48,672482	451,327518
600,00	0,900251	97,236853	58,647362	541,352638
700,00	0,900251	97,236853	68,622243	631,377757
800,00	0,900251	97,236853	78,597123	721,402877
900,00	0,900251	97,236853	88,572004	811,427996
1000,00	0,900251	97,236853	98,546884	901,453116

Figura 3.13 Tela de resultados da simulação

Olhando para a figura acima, nota-se que o programa mostra os resultados para o sistema em estado estável (Steady State Results), para o sistema no instante final de simulação (Results at time 1000) e para os instantes de tempo escolhidos pelo usuário. Os valores de **Availability** e **Capacity** coincidiram porque no instante 1000 o sistema já está em estado estável.

- O valor do critério **Availability** foi de 0.9, portanto existe a probabilidade de 90% de o sistema estar funcionando com 100% de capacidade no estado estável.
- O valor do critério **Capacity** foi de aproximadamente 97%, portanto essa é a capacidade média do sistema em estado estável.

- O valor do critério **Total Downtime** foi de aproximadamente 98, portanto em 98 das 1000 horas de simulação o sistema estava funcionando com capacidade menor que 100%, que foi a capacidade exigida como parâmetro no campo **Required Capacity** anteriormente, fazendo com que o sistema seja considerado **Down** caso esteja abaixo dessa capacidade.

- O valor do critério **Total Uptime** foi de aproximadamente 902, portanto em 902 das 1000 horas de simulação o sistema estava funcionando com capacidade de 100%, que foi a capacidade exigida como parâmetro no campo **Required Capacity** anteriormente, fazendo com que o sistema seja considerado **Up** caso esteja no mínimo com essa capacidade.

- O tempo médio entre a ocorrência de falhas consecutivas (MTBF) é de 111 horas, aproximadamente.

- O tempo médio para a ocorrência da primeira falha é de 100 horas, aproximadamente.

- O tempo médio para o sistema reparar uma falha é de 11 horas, aproximadamente.

Além dos cálculos para o sistema como um todo, o software também gera resultados específicos para cada estado. Na figura abaixo pode-se ver os valores do critério **Time Spent in State** para o estado inicial.

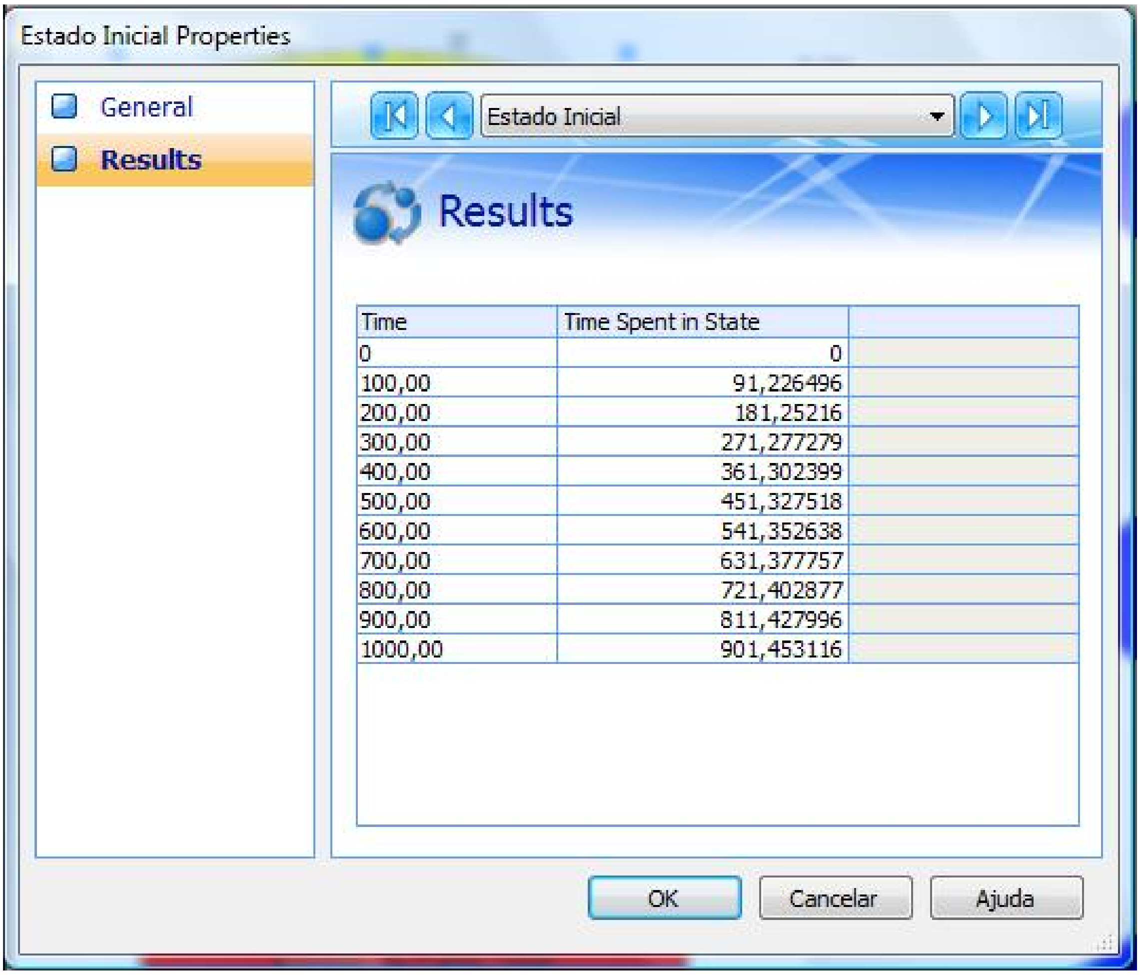


Figura 3.14 Tela de resultados para um estado

Pela Figura 3.14 nota-se que o usuário pode acompanhar o tempo que o sistema passou naquele estado durante a simulação. Nesse exemplo do estado inicial tem-se que o sistema passou aproximadamente 902 horas (901,45) no mesmo, assim funcionando com capacidade de 100%. Analisando cada estado, observou-se que o sistema passou **902** horas funcionando corretamente, **89** horas com capacidade de 75% (1 módulo de memória com falhas), **8** horas com capacidade de 50% (2 módulos de memória com falhas), **54** minutos com capacidade de 25% (3 módulos de memória com falhas) e **6** minutos em estado de falha total.

Além dos valores resultantes da simulação, o software também fornece a opção de gráficos, que pode ser ativada no menu superior do programa. Esses gráficos ilustram algumas características do sistema durante o tempo. Um deles é o gráfico **Availability** X **Time**, mostrado na figura abaixo:

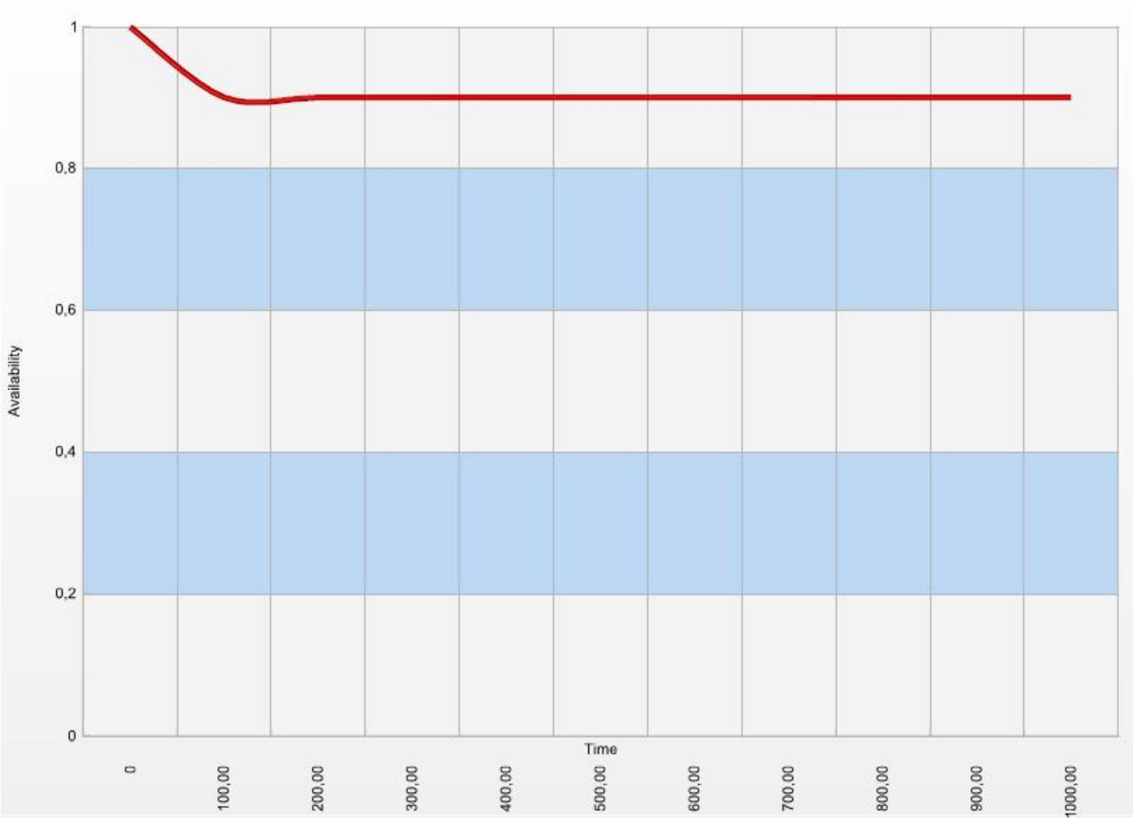


Figura 3.15 Gráfico **Availability** X **Time**

Analisando o gráfico da Figura 3.15, pode-se ver que o valor **Availability** começa a decair, ou seja, com a passagem do tempo a probabilidade de que o sistema esteja no estado de funcionamento correto vai diminuindo, até que o sistema entra em estado estável, assim mantendo o valor **Availability** em 0.9 independentemente do tempo.

Outro gráfico gerado pelo programa é o gráfico **Capacity** X **Time**, mostrado na figura abaixo:

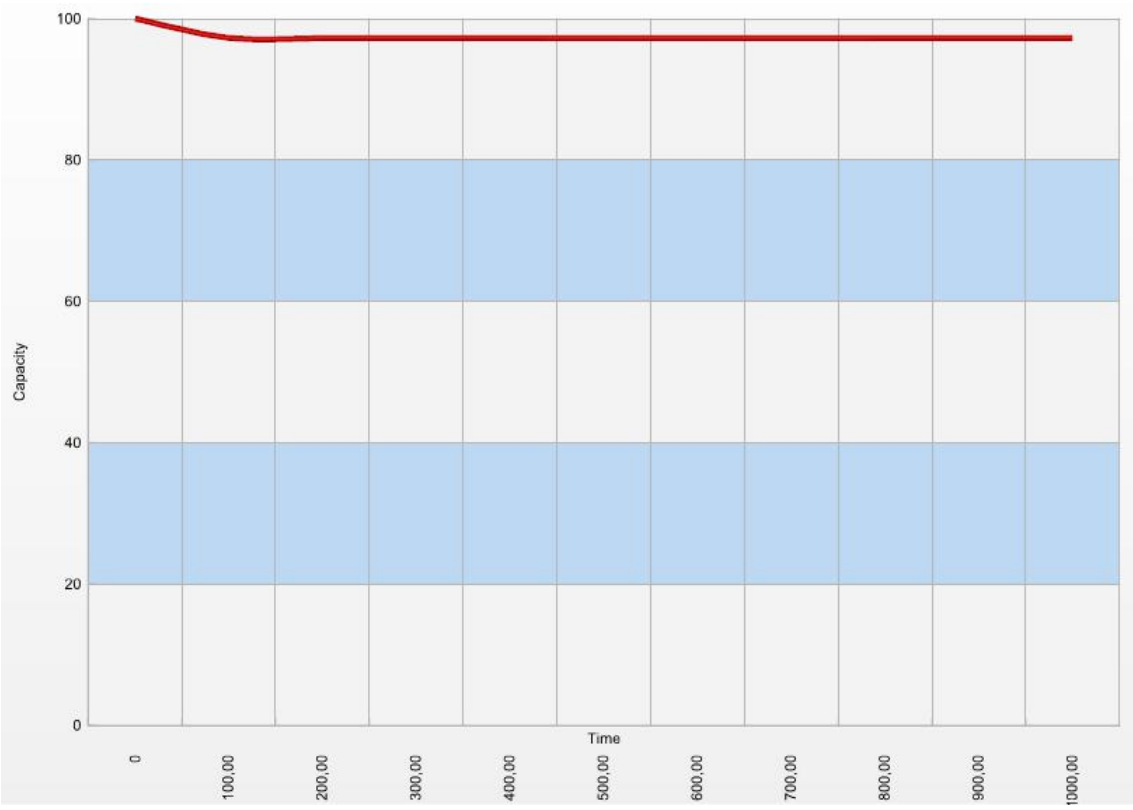


Figura 3.16 Gráfico **Capacity** X **Time**

Analisando o gráfico da Figura 3.16, pode-se ver que a capacidade média do sistema vai diminuindo com o tempo à medida que as falhas começam a ocorrer, depois atingindo o estado de equilíbrio e se mantendo no valor de 97%.

3.4 Soluções analíticas X Soluções de simulação

Ao modelar um sistema com base na teoria de cadeias de Markov, é necessário decidir que tipo de solução vai ser aplicada ao modelo, analítica ou de simulação, a não ser que os modeladores queiram usar os dois tipos de solução. Dependendo da situação, um método pode ser mais interessante que o outro. Para sistemas complexos e grandes, geralmente o uso de soluções de simulação é mais indicado, porque as soluções analíticas para esses casos muitas vezes requerem abstrações e aproximações inviáveis. As soluções de simulação têm a vantagem de permitir uma abordagem mais simples, não exigem a solução de sistemas de equações algumas vezes complexos, ao invés disso, simulam e exploram caminhos de execução da cadeia de Markov em questão, revelando as características do comportamento do sistema.

Apesar das vantagens citadas, as soluções de simulação têm desvantagens em relação às soluções analíticas. Por exemplo, se o modelo gerado tem eventos que acontecem raramente, mas que são críticos para o sistema, são necessários tempos de execução de simulação muito longos para esses eventos ocorrerem na simulação e serem analisados. Outra desvantagem é que as soluções de simulação na maioria das vezes são mais custosas e consumem mais tempo de análise do que as soluções analíticas. Assim, para sistemas de pequeno a médio porte, não muito complexos muitas vezes as soluções analíticas são mais atrativas.[2],[4],[8]

4 Considerações finais

Nesse trabalho foi mostrada uma maneira muito útil de avaliar características dinâmicas de sistemas computacionais, através da teoria de cadeias de Markov. É possível modelar um sistema computacional existente ou hipotético, com o objetivo de analisar características do comportamento desse sistema em funcionamento, em critérios como performance, disponibilidade e confiabilidade. Foi visto que os sistemas que podem ser modelados por essa teoria são aqueles que podem ser representados por estados e transições entre esses estados, e que têm adicionalmente a propriedade **memoryless**, ou seja, quando um sistema desse tipo está em um determinado estado presente, o estado futuro não depende dos estados passados, somente do estado presente. É comum dizer que nesse tipo de sistema toda a história do mesmo é resumida pelo estado atual.

No processo de modelagem, os modeladores precisam primeiramente abstrair características do sistema, assim podendo transformá-lo posteriormente em um modelo matemático formal que representa uma cadeia de Markov, a qual obedece ao processo de Markov. Após modelar o sistema, é preciso decidir qual tipo de solução utilizar para avaliá-lo, soluções analíticas ou soluções de simulação, como foi discutido no final do capítulo anterior. Após a solução ser aplicada e os resultados serem analisados, os interessados podem ter um conhecimento do comportamento dinâmico do sistema.

Concluindo, a modelagem de sistemas computacionais por cadeias de Markov é uma ferramenta poderosa para avaliar o comportamento de um sistema. Uma equipe que pretende fazer um projeto, como implementar uma rede ou um sistema distribuído, por exemplo, pode prever características comportamentais desses sistemas, sem precisar construir partes dos mesmos e testá-las, o que é custoso e demorado. Claro que o método de modelagem não é perfeito, existem diferenças entre o comportamento estimado pelo modelo e o sistema real em funcionamento, proporcionais à qualidade da modelagem, mas mesmo assim a análise por modelagem é muito útil, pois revela informações sobre o andamento do sistema e sobre os vários estados nos quais ele pode se encontrar.

5 Referências bibliográficas

[1] ALLEN, A. O. *Probability, Statistics, and Queueing Theory with Computer Science Applications.* Academic Press, Inc. 2. ed. 1990.

[2] BOLCH, G. ; GREINER, S. ; MEER, H. ; TRIVEDI, K. S. *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications.* John Wiley and Sons, Inc. 2. ed. 2006.

[3] CHANIN, R. ; DOTTI, F. L. ; FERNANDES, P. ; SALES, A. *Avaliação quantitativa de sistemas.* Faculdade de Informática da Pontifícia Universidade Católica do Rio Grande do Sul, 2005.

[4] HEIMANN, D. I. ; MITTAL N. ; TRIVEDI K.S. *Availability and Reliability modeling for computer systems.* Academic Press, Inc., 1990.

[5] MEYER, P. L. *Probabilidade: Aplicações à Estatística.* Tradução Ruy de C. B. Lourenço Filho. LTC - Livros Técnicos e Científicos Editora S. A. 2. ed. 2009.

[6] ROSS, S. M. *Introduction to Probability Models.* Academic Press, Inc. 6. ed. 1997.

[7] SANTOS, R. J. *Introdução ao Latex.* Departamento de Matemática da Universidade Federal de Minas Gerais, 2009.

[8] TRIVEDI, K. S. *Probability and Statistics with Reliability, Queueing and Computer Science Applications.* John Wiley and Sons, Inc. 2. ed. 2002.