A Utilização de Statecharts e Processo Markoviano de Decisão na Avaliação de Desempenho

Marcelino S. da Silva, Diego L. Cardoso, Carlos R. L. Francês, Nandamudi L. Vijaykumar and Solon V. de Carvalho

***Resumo*— Este trabalho propõe uma estratégia que associa uma especificação formal de alto nível, Statecharts, a um Processo Markoviano de Decisão. Os Statecharts são utilizados para propiciar uma representação clara do comportamento do sistema, incluindo as possíveis tomadas de decisões e os custos advindos de cada decisão. E o Processo Markoviano de Decisão analisa os efeitos em longo prazo de cada decisão. Como resultado dessa associação, permite-se que usuários do sistema, os quais não possuem um conhecimento profundo sobre o processo de avaliação de desempenho, possam participar de forma mais efetiva na etapa de modelagem, provendo informações que tornam o modelo mais coerente com o sistema real.**

***Palavras-Chave*— Avaliação de Desempenho, Processo Markoviano de Decisão e Statecharts.**

***Abstract*— This paper proposes a strategy that associates a high level formal specification, Statecharts, with a Markov Decision Process. Statecharts are used to represent in a clear manner the system behavior, including the possible decision choices and the costs incurred from each decision. And the Markov Decision Process is used to evaluate the long time effects of each decision. The result of this strategy is that the users of the system, without a good knowledge of the performance evaluation process, may participate in a more effective way in the modeling step, proving information to build a more realistic model.**

***Keywords*— Performance evaluation, Markov Decision Process and Statecharts.**

I. INTRODUÇÃO

O processo de modelagem com o enfoque de desempenho, na maioria de suas abordagens, não apresenta um equilíbrio entre uma boa especificação e uma solução viável para o sistema em estudo, sendo que, via de regra, a preocupação precípua se atem apenas ao método de solução. Essa certa "negligência" em relação à especificação pode estar associada ao perfil dos utilizadores da avaliação de desempenho, que, geralmente, possuem um conhecimento matemático apropriado. Dessa forma, para esse tipo de usuário, por exemplo, um conjunto de equações de um sistema linear pode ser tão claro quanto uma especificação gráfica em alto nível.

Se, entretanto, no processo de modelagem, há outras pessoas menos especializadas envolvidas (as quais são, em algum nível, usuários do sistema a ser modelado), poderia ser interessante dispor-se de uma forma de representação clara e abstrata o bastante, a ponto de apresentar o sistema, escondendo a complexidade que pode estar associada às soluções matemáticas que alicerçam a modelagem.

Marcelino S. da Silva, Diego L. Cardoso and Carlos R. L. Francês, Laboratório de Planejamento de Redes de Alto Desemepnho, Universidade Federal do Pará, Belém, PA, Brasil, E-mails: {marcelino,diego,rfrances}@ufpa.br. Nandamudi L. Vijaykumar and Solon V. de Carvalho¸ Laboratório Associado de Computação e Matemática Aplicada, Instituto Nacional de Pesquisas Espaciais - INPE, São José dos Campos, SP, Brazil, E-mails: {vijay,solon}@lac.inpe.br. Este trabalho foi parcialmente financiado pela CAPES, projeto PROCAD número 0226050.

Baseando-se nesse intuito de prover uma especificação clara e abstrata do sistema, foi desenvolvida a técnica de especificação formal Statecharts [1]. Essa técnica é uma extensão dos diagramas convencionais de estados e transições, na qual a principal característica dessa especificação situa-se na possibilidade de representar de forma clara e explícita os vários componentes paralelos e decomposições hierárquicas no sistema.

Para que os Statecharts possam ser utilizados no processo de avaliação de desempenho de sistemas, foi elaborado um método computacional para que, a partir da especificação em Statecharts do sistema, possa-se obter a Cadeia de Markov isomórfica a essa especificação [6].

Dando continuidade ao trabalho apresentado em [6], propõe-se uma estratégia que associa uma especificação em alto nível baseada em Statecharts a um Processo Markoviano de Decisão (PMD). A partir da especificação Statecharts podem ser estabelecidas indicações de quando o sistema deve optar por uma ou por outra decisão, assim como os custos de cada decisão tomada; possibilitando, desta forma, a construção de um Processo Markoviano de Decisão a Tempo Contínuo (PMDTC). De posse do PMDTC, pode-se obter a política ótima $R*$ que minimiza o custo médio, em longo prazo, do sistema.

O PMD tem sido amplamente utilizado nas mais diversas áreas do conhecimento, incluindo a modelagem e análise de sistemas de telecomunicações. Em [7] um PMD é utilizado para obtenção de um esquema de alocação de banda e gerência de buffer para redes sem fio do tipo ad hoc; em [3] é apresentado uma estratégia de transmissão de dados em uma comunicação ponto a ponto sem fio, na qual a modulação e a potência de transmissão do sinal são dadas por uma política obtida por um PMD.

O estudo de caso utilizado neste trabalho é um modelo de controle de máquinas clássico apresentado em [5]. Este modelo foi utilizado para que se possa comparar os resultados deste trabalho com os resultados de [5] para se verificar a confiabilidade dos resultado. Entretanto esta estratégia pode ser facilmente aplicada a sistemas de telecomunicações, como por exemplo, estudo de controles de admissão de chamadas para prover garantias de qualidade de serviço em redes móveis celulares, alocação de comprimentos de ondas em redes ópticas, entre outros.

II. Especificação Statecharts

Sistemas complexos envolvendo paralelismo, sincronização e interdependência de seus componentes ou subsistemas são nomeados de sistemas reativos. A principal característica desses sistemas consiste no fato de serem baseados em eventos, reagindo a estímulos tanto externos quanto internos.

Um dos principais problemas desse tipo de sistema consiste na dificuldade de descrever o seu comportamento de forma clara, concisa e livre de ambigüidades, uma vez que esse comportamento é dirigido por eventos complexos e inter-relacionados. Tal descrição torna-se inviável por meio de diagramas tradicionais de máquinas de estado, visto que pequenas variações no comportamento do sistema podem acarretar em grandes mudanças na representação do sistema e no crescimento exponencial do espaço de estados [1] e [2].

Para permitir uma representação clara de hierarquia, concorrência e interdependência entre componentes do sistema, [1] apresenta uma técnica de especificação formal denominada Statecharts. Os principais elementos desta especificação são apresentados a seguir.

Estados são usados para descrever componentes (e suas possíveis situações) de um determinado sistema. Os estados de um Statecharts podem ser classificados em dois grupos: básicos e não-básicos. Os estados básicos são aqueles que não possuem sub-estados. Já os não-básicos são decompostos em sub-estados. Essa decomposição pode ser de dois tipos: XOR ou AND. Se a decomposição é do tipo XOR, então esse estado do sistema não poderá estar em mais de um sub-estado simultaneamente. Entretanto, se a decomposição é do tipo AND, esse estado poderá encontrar-se em mais de um sub-estado simultaneamente.

Eventos são importantes na especificação Statecharts, pois são esses elementos que interferem na dinâmica do sistema. Opcionalmente, a um evento pode ser anexada uma condição (entre colchetes *[]*), também chamada de condição-guarda, de maneira que o evento só ocorrerá (ou só irá interferir no sistema) se satisfizer esta determinada condição. Os Statecharts classificam os eventos em externos e internos. Os eventos externos são aqueles que devem ser explicitamente estimulados, enquanto os internos são detectados automaticamente pelo sistema e são imediatamente estimulados sem um estímulo explícito.

Para se utilizar os Statecharts como ferramenta de análise de desempenho, os eventos externos (explicitamente estimulados) carregam uma informação estocástica com alguma distribuição de probabilidade, por exemplo, taxas de chegada, taxas de serviço, etc. Em particular, para se associar a especificação Statecharts a uma Cadeia de Markov, o tempo entre ocorrências de um evento deve ser exponencialmente distribuído. Os internos não terão esta informação estocástica associada, ou seja, o tempo de espera para a ocorrência da transição é zero, quando estimulados automaticamente.

O elemento ação é utilizado para descrever os efeitos do paralelismo em Statecharts (a influência de um estado em outro estado, dado que esses dois estados são paralelos entre si).

As transições são a representação gráfica para denotar uma mudança de estado dentro do sistema. Rótulos podem ser acrescentados às setas para prover algum significado adicional.

Para ilustrar as definições apresentadas, observa-se o sistema modelado em Statecharts apresentado na Figura 1. Considera-se um sistema que possui duas máquinas idênticas atendendo a uma mesma fila de requisições. A cada instante de tempo estas duas máquinas podem ser ligadas ou desligadas através dos eventos *lig1* e *desl1*, para a máquina M1, e *lig2* e *desl2* para a máquina M2, conforme a "vontade" do controlador.

Nesse sistema identificamos quatro componentes básicos que atuam em paralelo: FONTE (componente que representa a chegada de requisições ao sistema), FILA (representa o número de clientes no sistema, tanto os sendo atendidos quanto os em espera), M1 e M2 (representam as duas máquinas do sistema). Desta forma, o estado raiz do sistema (denominado FILA COM 2 SERVIDORES) pode ser modelado como um estado não-básico do tipo AND, no qual FONTE, FILA, M1 e M2 são sub-estados desse estado raiz. A representação gráfica de paralelismo é feita separando-se os estados paralelos por uma linha pontilhada.

Como pode ser observado pela Figura 1, os quatro componentes paralelos comunicam-se através de ações, as quais disparam os eventos internos *inc* e *dec* em FILA, que incrementam e decrementam a fila, respectivamente.

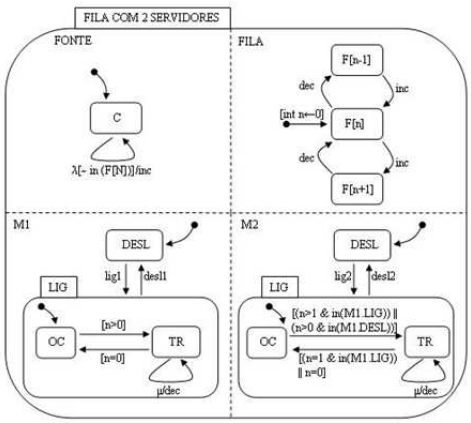

Fig. 1. Modelo em Statecharts de uma fila e dois servidores

Observa-se que os termos *estado* e *ação* são utilizados tanto na descrição de PMD quanto na descrição de Statecharts. Entretanto, em cada uma dessas modelagens, cada termo possui uma interpretação diferente.

O termo *estado* é utilizado em PMD para denotar a condição em que o sistema, como um todo, se encontra. Já em Statecharts esse termo possui uma utilização mais ampla, referindo-se não apenas a condição em que o sistema encontra-se, mas também a componentes específicos do sistema. Para retirar essa ambigüidade será utilizado o termo *estado* quando se estiver referindo ao estado de um Statecharts e para os PMD substituiremos o termo *estado* por *configuração* (por exemplo, "espaço de estados S" será referenciado como "espaço de configurações S").

O termo *ação*, o qual em PMD indica qual decisão foi tomada e em Statecharts indica como a ocorrência de um evento em um estado afeta outro estado paralelo, continuará sendo utilizado como especificado em Statecharts, enquanto

que para o PMD será utilizado sempre o termo *decisão* (por exemplo, substituiremos "a ação a foi tomada no estado i" por "a decisão a foi tomada no estado i").

III. Construção de um PMDTC a partir de Statecharts

Nessa seção são apresentados uma extensão ao Statecharts para que este possa ser convertido em um PMDTC e o método computacional para realizar a conversão.

A. Adaptações necessárias ao Statecharts

Para se obter um PMDTC a partir de um Statecharts, faz-se necessário a inclusão de um novo elemento a especificação Statecharts. Este novo elemento, denominado *estado de decisão*, indica que quando o sistema encontra-se neste estado, deve-se escolher uma e apenas uma decisão a ser tomada, a qual leva, de forma instantânea, o sistema para um novo estado. Como notação gráfica para esse novo elemento, sugere-se a utilização de um círculo com a letra D (decisão) no centro, no qual as transições (eventos) que levam o sistema a este estado de decisão indicam que uma decisão deve ser tomada após a ocorrência desses eventos e as transições que saem desse estado de decisão representam, cada, uma possível decisão e o estado de chegada de cada transição indica qual estado é alcançado quando a decisão referente a essa transição é tomada. A Figura 2 apresenta um exemplo dessa notação.

A Figura 2 ilustra um componente A de um dado sistema. Quando o sistema se encontra nesse componente (estado A), um evento λ pode ser disparado e a ocorrência desse evento leva o sistema a um estado de decisão no qual se deve optar por tomar a decisão $< 0 >$ que dispara a ação x ou tomar a decisão $< 1 >$ que dispara a ação y.

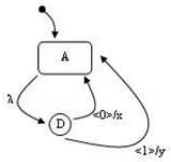

Fig. 2. Exemplo de utilização do estado de decisão

No exemplo apresentado na Figura 3 o sistema pode optar por três decisões: decisão $< 0 >$, a qual desliga as duas máquinas; decisão $< 1 >$, que liga apenas a máquina M1; e decisão $< 2 >$, que liga as duas máquinas. Observa-se que não existe a opção de desligar M1 e ligar M2, isto ocorre porque, como se considerou que as duas máquinas são idênticas com os mesmos custos e taxas de atendimento, tomar a decisão de desligar M1 e ligar M2 seria redundante com decidir por ligar M1 e desligar M2.

Para tratar o sistema como um PMDTC, faz-se necessário que os eventos que levam o sistema ao estado de decisão sejam eventos estocásticos e que a distribuição de probabilidade do tempo entre ocorrências de um evento seja exponencial.

Para a descrição dos custos acarretados ao sistema, sugere-se a utilização de uma tabela de custos em conjunto com a especificação em Statecharts. A escolha por utilizar uma tabela para especificar os custos, em detrimento de incluir tais custos diretamente na especificação Statecharts, foi feita para evitar "sobrecarregar" a especificação Statecharts, mantendo-se a especificação clara e objetiva como o que foi proposto inicialmente [1] e [2].

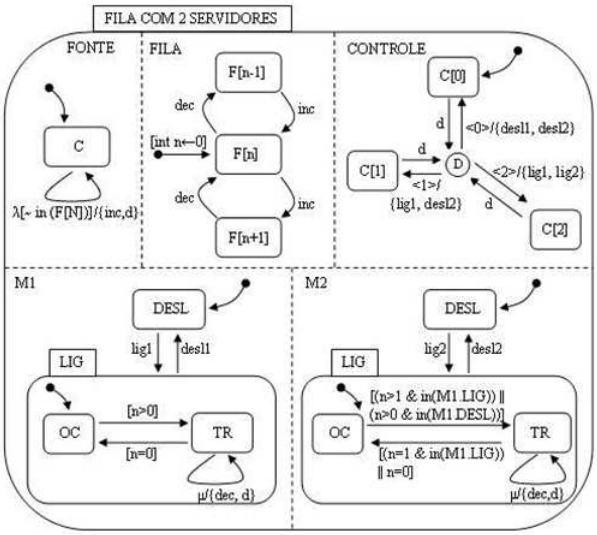

Fig. 3. Modelo em Statecharts de um sistema com dois servidores e uma fila utilizando um estado paralelo para controlar as máquinas

TABELA I

Tabela de custos para o sistema com uma fila e dois servidores

Custos de Permanência		
Estado		**Custo**
M1.LIG		cm_1l
M2.LIG		cm_2l
F(n)		$n.cr$
Custos de Transições		
Estado Sucessor	**Estado Destino**	**Custo**
M1.DESL	M1.LIG	clm_1
M2.DESL	M2.LIG	clm_2
M1.LIG	M1.DESL	cdm_1
M2.LIG	M2.DESL	cdm_2
Custos Particulares		
$-gf(tempo(M1.TR)\mu + tempo(M2.TR)\mu))$		

Para montar a tabela de custos, dividem-se os custos em três tipos básicos: (1) custos de permanência; (2) custos de transições; (3) custos particulares. O primeiro tipo de custo refere-se a custos que são incididos ao custo total devido à permanência do sistema em um determinado estado. O segundo tipo trata dos custos relacionados às transições entre estados. Estes dois primeiro tipos de custos possuem campos específicos na tabela por serem muito comuns em PMD. Entretanto, em alguns casos particulares, o custo de cada decisão é calculado como uma função de diversos fatores, sendo necessário que o analista especifique-o de forma explicita. Nesse caso, utiliza-se o campo da tabela dedicado aos custos particulares. A Tabela I demonstra a tabela de custos para o sistema da Figura 3.

Pela Tabela I identifica-se que o custo total por unidade de tempo composto por oito componentes:

1) Custo de manter a máquina M1 ligada: quando o sistema estiver no estado M1.LIG, será adicionado ao custo total um custo $cm1l.T$, onde $cm1l$ representa o custo por unidade de tempo de manter a máquina M1 ligada e T representa o tempo de permanência nesse estado;
2) Custo de manter a máquina M2 ligada: semelhante a manter M1 ligada;
3) Custo para manter cada requisição no sistema: toda vez que existirem n requisições no sistema (o sistema está no estado F(n)) será adicionado ao custo total um custo $n.cr.T$, onde cr é o custo para manter uma requisição no sistema por unidade de tempo.
4) Custo de ligar a máquina M1: quando o estado M1.DESL fizer parte da configuração atual do sistema (configuração i), o sistema poderá passar para a configuração j em que o estado M1.LIG faz parte dessa configuração com probabilidade $p_{ij}(a)$, dado que a é a decisão tomada na configuração i. Então o custo acarretado por ligar a máquina M1 será calculado como $clm_1 \sum_{j \in S} p_{ij}(a)$;
5) Custo de ligar a máquina M2: semelhante ao custo de ligar M1;
6) Custo de desligar a máquina M1: semelhante ao custo de ligar, consideranso-se que o estado muda de M1.LIG para M1.DESL;
7) Custo de desligar a máquina M2: semelhante ao custo de desligar M1;
8) Ganho por finalizar uma requisição: quando a configuração atual do sistema (configuração i) indicar que o sistema está no estado F(n), o sistema poderá passar para a configuração j em que o estado F(n-1) faz parte dessa configuração com probabilidade $p_{ij}(a)$, dado que a é a decisão tomada na configuração i. Então o custo acarretado por realizar essa transição é calculado como $-gf \sum_{j \in S} p_{ij}(a)$. Observa-se que esse custo possuí o sinal negativo, isto indica que na verdade esse valor não é um custo e sim um ganho;

Tendo-se especificado o sistema em Statecharts com a extensão proposta e a tabela de custos, é então possível construir um PMDTC para encontrar a política ótima $R*$ do sistema. O algoritmo para construção do PMDTC é apresentado na seção seguinte.

B. Algoritmo para construção do PMDTC

Este trabalho propõe que a partir de uma especificação em Statecharts com as devidas indicações de quando o sistema deve optar por uma ou outra decisão e os custos acarretados ao sistema quando uma decisão é escolhida, é possível construir um Processo Markoviano de Decisão a Tempo Contínuo para obter-se a política ótima $R*$ que minimiza o custo médio em longo prazo do sistema.

Visto que na literatura já existem vários métodos eficientes para a obtenção da política $R*$ [5], [4], de forma particular nesse trabalho será utilizado o Algoritmo de Iteração de Valores [5], o problema restringe-se a encontrar as configurações adequadas do sistema para se montar o PMDTC, as decisões que podem ser tomadas em cada configuração do PMDTC e as taxas de saída de cada configuração do PMDTC quando certa decisão foi escolhida.

Para montar um PMDTC a partir de Statecharts, primeiramente monta-se um grafo, no qual cada nó desse grafo representa uma configuração do sistema. Admite-se que cada configuração (nó do grafo) pode ser de dois tipos: (1) uma configuração na qual o sistema não se encontra em um estado de decisão; e (2) uma configuração onde o sistema encontra-se em um estado de decisão (esse nó será marcado como "configuração de decisão"). Cada aresta do grafo será direcionada, representada por uma seta, no qual cada seta pode ser também de dois tipos: (1) uma seta estocástica que indica que uma transição é feita quando um evento estocástico ocorre, isto é, o tempo de espera para que essa transição seja disparada é dado por uma distribuição de probabilidade exponencial com média $1/\lambda$, onde λ é o rótulo da seta e indica a taxa de ocorrência desse evento; e (2) uma seta de decisão que indica qual deve ser o nó destino do grafo quando a decisão $< a >$ é tomada, onde $< a >$ é o rótulo dessa seta (admite-se nesse caso que, quando uma decisão é tomada, a transição é executada de forma instantânea).

O grafo será montado segundo a condição de que cada nó do grafo representará uma configuração do sistema, no qual a saída dessa configuração somente pode ser feita ou por uma tomada de decisão ou pela ocorrência de um evento estocástico. Essa condição implica em dizer que as configurações em que existem eventos internos habilitados não farão parte do grafo. Isso ocorre porque nesses casos o tempo de permanência nessas configurações é zero, visto que, ao encontrar-se nessas configurações, automaticamente as transições que denotam os eventos internos habilitados serão disparadas.

A partir dessa condição, utiliza-se o seguinte algoritmo para montar o grafo:

geraGrafo(Statecharts **sc**, Grafo **g**)
Início
  Obtém-se a configuração inicial **c** de **sc**
  Estando na configuração **c**, se existirem eventos internos habilitados, dispara-se essas transições e continua-se reagindo até encontrar uma configuração em que não existem mais eventos internos habilitados
  Adiciona-se essa nova configuração ao grafo **g** e marca-se esse nó como "não expandido"
  Enquanto existir um nó em **g** marcado como "não expandido", executa-se o seguinte laço
  Início
    fonte é o nó marcado como "não expandido"
    Se **fonte** for uma configuração de decisão então
    Início
      Monta-se o conjunto **ds**, formado pela combinação das decisões de cada estado de decisão da configuração **fonte**
      Para cada combinação **d**∈**ds** faça
      Início
        Encontra-se a configuração **destino** alcançada quando a combinação de decisões **d** é disparada a partir da configuração **fonte**
        Estando na configuração **destino**, se existirem eventos internos habilitados, dispara-se essas transições e continua-se reagindo até encontrar uma configuração em que não existem mais eventos internos habilitados e atribui-se essa nova configuração a **destino**
        Se **destino** ainda não fizer parte do conjunto de nós de **g**, adiciona-se destino a **g** e marca-se destino como "não expandido"
        Adiciona-se em **g** uma seta de decisão saindo de **fonte** e chegando a **destino** com o rótulo "**d**"
      Fim
    Fim

Se *fonte* NÃO for uma configuração de decisão então
Início
Monta-se o conjunto **es** de todos os eventos estocásticos que podem ocorrer enquanto o estiver na configuração **fonte**
Para cada evento estocástico **e**∈**es** faça
Iníco
Encontra-se a configuração **destino** alcançada quando o evento **e** é disparada a partir da configuração **fonte**
Estando na configuração **destino**, se existirem eventos internos habilitados, dispara-se essas transições e continua-se reagindo até encontrar uma configuração em que não existem mais eventos internos habilitados e atribui-se essa nova configuração a **destino**
Se **destino** ainda não fizer parte do conjunto de nós de **g**, adiciona-se **destino** a **g** e marca-se **destino** como "não expandido"
Se já existir uma seta estocástica em **g** saindo de **fonte** e chegando a **destino**, então adiciona-se "**+e**" ao rótulo da seta, senão adiciona-se a **g** uma seta estocástica saindo de **fonte** e chegando a **destino** com rotulo "**e**"
Fim
Fim
Marca-se fonte como "expandido"
Fim

Fim

A Figura 4 demonstra o grafo gerado para o sistema especificado em Statecharts da Figura 3, no qual as setas pontilhadas representam as setas de decisão e as setas contínuas representam as setas estocásticas. Considerou-se o tamanho máximo da fila igual a três ($N = 3$).

Quando se constrói um PMDTC admite-se que todas as vezes que um evento estocástico ocorre é necessário que uma decisão seja tomada [4]. Para o exemplo da Figura 3, os eventos estocásticos são chegada de clientes, λ, e término de atendimento, μ. Sendo assim, sempre que ocorrer um desses eventos, uma decisão deve ser tomada.

O espaço de configurações do PMDTC será formado pelo conjunto de configurações do sistema, no qual pelo menos um dos estados é um estado de decisão. Isto indica que cada configuração do PMDTC representa a configuração que o sistema encontra-se após a ocorrência de um evento estocástico e antes de uma decisão ter sido tomada. As possíveis decisões de cada configuração serão dadas pelas setas de decisões que saem do nó que representa cada configuração.

Para se encontrar as taxas de saída de uma configuração do PMDTC dada uma certa decisão, observa-se o nó que representa essa configuração, a decisão tomada e o nó alcançado por essa decisão. Esse último nó representa a configuração em que o sistema irá de fato atuar, ou seja, se o sistema encontra-se na configuração representada pelo nó **fonte** e a decisão **d** é escolhida, então o sistema irá operar na configuração representada pelo nó **auxconf**. Dessa forma, as taxas de saída da configuração representada pelo nó **fonte**, quando a decisão **d** é tomada, serão indicadas pelas setas estocásticas que saem de **auxconf**.

Da mesma forma, cada seta estocástica saindo de **auxconf** irá indicar qual configuração **destino** é alcançada quando o sistema encontra-se na configuração **fonte** e a decisão **d** é escolhida.

O algoritmo que executa esse processo de conversão do grafo para o PMDTC é mostrado a seguir:

geraPMDTC(Grafo **g**, PMDTC **pmd**)
Início

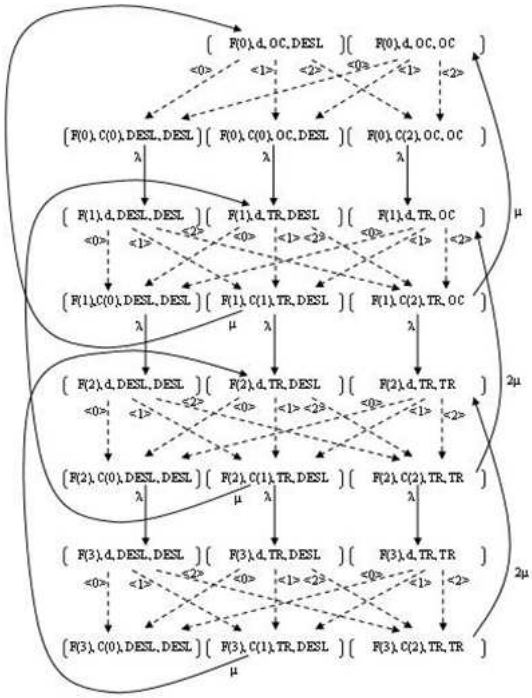

Fig. 4. Grafo do sistema com dois servidores e uma fila

Para cada nó do grafo **g** que representa uma configuração de decisão faça
Início
fonte é o nó marcado representando a configuração de decisão
Monta-se o conjunto **ds**, formado pelas setas de decisões saindo do nó **fonte**
Para cada seta de decisão **d**∈**ds** faça
Início
Encontra-se o nó **auxconf** alcançado pela seta **d** a partir do nó **fonte**
Monta-se o conjunto **es** de todas as setas estocásticas saindo do nó **fonte**
Para cada seta estocástica **e**∈**es** faça
Início
Encontra-se o nó **destino** alcançado pela seta **e** a partir do nó **auxconf**
Se a configuração representada pelo nó **fonte** ainda não faz parte do espaço de configuração do **pmd**, então se adiciona **fonte** ao **pmd**
Se a configuração representada pelo nó **destino** ainda não faz parte do espaço de configuração do **pmd**, então se adiciona **destino** ao **pmd**
Adiciona-se ao **pmd** a decisão **d** para a configuração **fonte** e adiciona-se que se a decisão **d** é tomada na configuração **fonte**, o sistema muda para o estado **destino** com taxa **e**
Fim
Fim
Fim

Fim

Para obter a política ótima $R*$ que minimiza o custo médio em longo prazo do PMDTC obtido pelo algoritmo demonstrado, aplica-se o AIV [5]. Os custos são calculados utilizando-se a tabela de custos.

Após executar o AIV obtêm-se os seguintes resultados: (1) política ótima que indica qual decisão deve ser tomada quando o sistema encontra-se na configuração **c**; (2) custo médio total em longo prazo e custos individuais conforme descriminado na tabela de custos; e (3) as probabilidades

(em regime estacionário) do sistema estar em cada uma das configurações e em cada estado. Um exemplo numérico é apresentado na próxima seção.

IV. Resultados Numéricos

Para verificar a confiabilidade do PMDTC gerado pelo método apresentado neste trabalho, utilizou-se o sistema da Figura 3. Este sistema foi escolhido por ser um exemplo clássico e didático, o qual foi apresentado em [5]. Os resultados obtidos neste trabalho são comparados com os resultados obtidos em [5], no qual o autor modelou o sistema diretamente como um PMDTC, além de realizar um tratamento matemático para reduzir o número de configurações do PMDTC.

Considera-se que o sistema possui dez máquinas disponíveis para atender as requisições ($s = 10$) e que a cada época de decisão o sistema pode tomar uma decisão $d = \{0, 1, 2, ...10\}$, na qual essa decisão indica o número de máquinas que devem permanecer (ou devem ser) ligadas. A Taxa de chegadas no sistema é de sete requisições por unidade de tempo ($\lambda = 7$), os dez servidores são idênticos e possuem uma taxa de atendimento de uma requisição por unidade de tempo ($\mu = 1$), o custo de manter cada máquina ligada é de 30 por unidade de tempo $(cmxl = 30)$, cada requisição que permanece no sistema gera um custo $h = 10$ por unidade de tempo e toda vez que se liga ou desliga uma máquina um custo $clmx = cdmx = 10$ é acarretado ao sistema. Nesse exemplo não será considerado o ganho por finalizar uma requisição $(gf = 0)$. O tamanho máximo considerado para a fila foi de $N = 100$.

Comparando os resultados obtidos nesse trabalho com os resultados obtidos em [5], tem-se que o custo médio em longo prazo por unidade de tempo é de 319,65 e 319,4, para o custo calculado pelo método apresentado nesse trabalho e o custo calculado em [5], respectivamente. Observa-se um erro de apenas 0,0078 por cento quando se considera $N = 100$ em detrimento de utilizar uma formulação matemática como a de [5], naqual não foi necessário assumir um limite N para o tamanho da fila.

TABELA II
POLÍTICA ÓTIMA PARA O SISTEMA COM DEZ SERVIDORES E UMA FILA

Requisições	s1	s2	Requisições	s1	s2
i=0	0	3	i=7	5	8
i=1	1	4	i=8 ou 9	6	9
i=2	2	4	i=10	7	10
i=3	2	5	i=11 ou 12	8	10
i=4	3	6	i=13 ou 14	9	10
i=5	4	6	$i \geq 15$	10	10
i=6	5	7	-	-	-

Quando se compara a política $R*$ obtida pelo método apresentado nesse trabalho com a política $R*'$ obtida em [5], observa-se que as duas políticas são idênticas. Essa política é apresentada na Tabela II, na qual os valores $s1$ e $s2$ indicam os limites máximo e mínimo de quantas máquinas devem permanecer ligadas a cada instante de decisão. Isto é, se em uma época de decisão m máquinas estão ligadas e o número de requisições no sistema é i, então, se $m < s1$, deve-se mudar para a configuração com $s1$ máquinas ligadas, se $s1 \leq m \leq s2$, o sistema deve permanecer com as m máquinas ligadas e se $m > s2$, então o número de máquinas ligadas deve ser reduzido para $s2$.

V. Comentários Finais

Este artigo apresentou uma estratégia que associa dois aspectos fundamentais em avaliação de desempenho de sistemas complexos: (i) a especificação formal e com um alto grau de abstração provida pelos Statecharts, (ii) a capacidade de apresentar cenários, atribuindo custos às transições de estados, utilizando-se Processo Markoviano de Decisão.

A utilização da técnica Statecharts apresenta as vantagens de possibilitar a representação clara de atividades e/ou componentes paralelos do sistema que se comunicam; permite o uso de hierarquia de estados para se "esconder" aspectos não essenciais à avaliação de desempenho realizada; e possibilita que usuários do sistema, menos familiarizados com as técnicas de avaliação de desempenho, possam ter um entendimento mais claro e objetivo do modelo implementado. E dessa forma, possam sugerir mudanças que tornem o modelo mais coerente com o sistema real.

Já as vantagens de se utilizar o Processos Markovianos de Decisão consistem no fato de que este permite que seja realizada uma análise em longo prazo do sistema a respeito dos custos acarretados ao sistema devido às decisões escolhidas; pode-se obter uma política ótima que norteia as tomadas de decisões ao longo do tempo; e o próprio fundamento matemático dos PMD, o qual estabelece que o próximo estado do sistema depende apenas do estado atual, permite a representação do sistema através de técnicas de especificação de alto nível, neste caso particular, Statecharts.

Referências

[1] D. Harel, *Statecharts: a visual formalism for complex systems*. Science of Computer Programming, Vol. 8, p. 231-274, 1987.

[2] D. Harel, A. Pnueli, J. Schmidt and R. Sherman, *On the formal semantics of Statecharts*. Proceedings of IEEE Symposium On Logic In Computer Science, Ithaca, USA. p.54-64, 1987.

[3] C. Pandana and C. K. J. R. Liu, *A near-Optimal Reinforcement Learning Scheme for Energy Efficient pointto-point Wireless Communications*, in IEEE Global Telecommunications Conference, 2004 - GLOBECOM '04, pp. 763-767, 2004.

[4] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

[5] H. C. Tijms, *Stochastic models: an algorithmic approach*, John Wiley & Sons, 1994.

[6] N. L. Vijaykumar, *Statecharts: Their use in specifying and dealing with Performance Models*, Tese de Doutorado, Instituto Tecnológico de Aeronáutica (ITA), São José Dos Campos, Brasil, 1999.

[7] D. Yagan, and C. K. Tham, *Self-Optimizing Architecture for QoS Provisioning in Differentiated Services*, in ICAC '05: Proceedings of the Second International Conference on Automatic Computing, pp. 346-347, 2005.