ALLOS - Uma ferramenta para solucionar modelos de redes de filas usando cadeias de Markov

Stanley R. de M. Oliveira
EMBRAPA/CNPTIA
Rod. D. Pedro I - Km 143,6 (SP65)
13.089-500 - Campinas, SP
Fone (0192) 40-1073, FAX (0192) 40-2007
stanley@dsc.ufpb.br

Maria Izabel C. Cabral
DSC/CCT/UFPB
izabel@dsc.ufpb.br

Edilson Ferneda
DSC/CCT/UFPB
edilson@dsc.ufpb.br

Marcos A. G. Brasileiro
DEE/CCT/UFPB
mbrasile@brufpb2.bitnet

Universidade Federal da Paraíba
Avenida Aprígio Veloso s/n - Bodocongó
58.109-970 Campina Grande, PB

RESUMO

Este trabalho apresenta uma ferramenta amigável para solucionar modelos de redes de filas usando cadeias de Markov. O processo de geração dos espaços de estados dos modelos a serem solucionados é feito automaticamente a partir de um estado inicial e de regras que descrevem o comportamento do modelo. O uso da ferramenta direciona-se, principalmente, aos sistemas que exibem contenção de recursos, em particular sistemas de redes de computadores.

ABSTRACT

This paper presents a friendly tool to solve models for networks of queues using Markov chains. The state space generation process for an especific model is automaticaly performed from a given initial state, according to rules that describe the behavior of that model. The use of this tool is adequate for solving systems models which exhibit resource contention, in particular, computer networks systems.

1. Introdução

A complexidade da nova geração de sistemas altamente concorrentes impõe o desenvolvimento de sofisticadas ferramentas de modelagem para a análise de desempenho de tais sistemas [1]. A modelagem e a análise de sistemas de computação têm sido enfocadas, com bastante ênfase, por parte de pesquisadores e projetistas que desejam entender e predizer o comportamento destes sistemas [2].

Um dos mecanismos adotados na análise de desempenho de um sistema é a obtenção de uma abstração que englobe o seu comportamento. Essa abstração é chamada de modelo do

sistema. Um modelo é uma visão simplificada do sistema em estudo, porém projetado com o propósito de capturar o máximo do comportamento real do sistema. Em muitos casos, modelos possuem certas vantagens óbvias sobre medidas feitas em um sistema real. Algumas vezes, um modelo é a única alternativa prática se o sistema ainda não foi concebido **[2,3]**.

Existem sistemas que apresentam contenção de recursos, como por exemplo, sistemas de computação e de comunicação, redes de computadores, controle de processos, sistemas de manufatura e sistemas de tráfego, os quais podem ser modelados através do paradigma de redes de filas **[4,5]**. Uma rede de filas é um sistema em que se considera a possível existência de múltiplas filas operando assíncrona e concorrentemente, interconectadas conforme uma topologia específica. Cada sistema de fila é constituído por clientes, um conjunto de servidores e uma ordem pela qual esses clientes são atendidos **[4,6]**.

A solução de modelos de redes de filas pode não ser uma tarefa fácil, sendo restrita apenas àqueles que possuem conhecimento das técnicas disponíveis na literatura. Para solucionar tais modelos, podem-se utilizar técnicas analíticas, que são mais econômicas, eficientes, contudo muitas vezes exigem a simplificação do modelo em virtude de sua complexidade. Outra alternativa são técnicas aproximadas, onde o custo computacional pode ser um fator limitante.

Uma das alternativas analíticas para a solução de redes de filas é o uso de processos markovianos. Um processo é markoviano se, dado o estado presente, o comportamento futuro do processo é estatisticamente independente do seu passado, isto é, no estado presente a história do sistema é sumarizada **[2,6]**. Essa classe de modelos é a mais amplamente usada na análise e modelagem de sistemas de computação/comunicação, e é suficientemente adequada na maioria dos casos.

Tradicionalmente, utilizavam-se na modelagem de sistemas de redes de filas as técnicas analíticas, quando possível, ou empregavam-se linguagens de propósito geral, tais como GPSS **[7]**, SIMULA **[8]**. Percebe-se hoje, uma tendência no uso de ambientes integrados que fornecem soluções analíticas e aproximadas e, além disso, apresentam capacidade de desenvolvimento de modelos poderosos, permitindo aos usuários a construção, modificação e a representação de modelos do mundo real. Exemplos de alguns destes ambientes é o SAVAD **[9,10]**, RESQ **[11]**, BONes e OPNET **[12]**.

Para a construção de tais ambientes, propõem-se técnicas estruturadas, facilidade de interação com o ambiente computacional, componentes de software reutilizáveis e facilidade de extensão de bibliotecas de módulos de software **[13,14]**.

O SAVAD (Sistema de AVAliação de Desempenho de Modelos de Redes de Filas) é um projeto multidisciplinar que envolve os Grupos de Redes de Computadores e de Inteligência Artificial do CCT/UFPB. Este sistema foi implementado usando as Linguagens de

Programação *C++* **[14]** e *PROLOG* **[15]** em sistemas de microcomputadores compatíveis com IBM-PC.

O SAVAD é um ambiente inteligente e integrado para modelagem e avaliação de desempenho de redes de filas. Ele é, basicamente, constituído de três módulos: *Interface*, *Núcleo do sistema especialista* e *Solução*. A abordagem orientada a objetos é utilizada no projeto e implementação deste sistema, uma vez que esta abordagem apresenta vantagens significativas, tais como a modularidade, a extensibilidade e a portabilidade.

A partir de um modelo de redes de filas proposto pelo usuário, o SAVAD procura uma solução para o modelo e fornece as medidas de desempenho solicitadas. Para tanto, o sistema deve escolher adequadamente as técnicas utilizadas, sejam elas exatas ou aproximadas. Entre as técnicas exatas, ressaltamos neste artigo, a ferramenta ALLOS, que gera automaticamente os estados dos modelos markovianos, solucionando-os, para posteriormente, apresentar as medidas de desempenho solicitadas. O processo de geração do espaço de estados de um modelo a ser solucionado é feito a partir de um estado inicial e das regras que descrevem o comportamento deste modelo.

A opção pelo uso de cadeias de Markov se dá pelo fato de que modelos markovianos têm causado impacto nos trabalhos de pesquisa concernentes à área de avaliação de desempenho de sistemas em comparação com técnicas aproximadas de simulação, visto que simulações são programas que, comparados com modelos matemáticos, são dispendiosos para construir, dificeis de serem validados, e caros em termos de ciclos de execução em CPU; enquanto que modelos markovianos necessitam de menos tempo para a sua construção e são menos susceptíveis a erros **[2]**. Outro fator relevante é que um modelo markoviano pode ser alterado (e.g., truncando o espaço de estados) de tal forma que este se torne mais tratável.

Neste artigo descrevemos a ferramenta *ALLOS* e apresentamos um exemplo do seu domínio de aplicação, na área de redes de computadores. Na seção 2 fazemos um resumo suscinto da interface de *ALLOS* e destacamos suas facilidades. Na seção 3 destacamos o processo de modelagem usando *ALLOS*, enfocando os procedimentos de geração automática de estados. Um exemplo numérico é mostrado na seção 4, com ênfase na solução e validação da ferramenta e, finalmente na seção 5, apresentamos nossas conclusões.

2. Interface de *ALLOS*

A interface da ferramenta *ALLOS* permite uma interação amigável entre o usuário e o sistema, de forma que o usuário que não necessita ser um especialista em modelagem e avaliação de desempenho de sistemas, auxiliando o usuário na construção de modelos.

Inicialmente, um usuário fornece o nome do modelo que desejar especificar. Caso o modelo já exista, ele pode ser facilmente alterado. Após fornecer o nome do modelo, a

interface interage com o usuário, solicitando deste os elementos de modelagem (descritos na seção 3.1. deste artigo) que serão utilizados, com seus respectivos atributos. Se algum destes elementos de modelagem já estiver definido, a interface apresenta-o com os seus parâmetros preeenchidos.

Os componentes utilizados na descrição dos modelos de redes de filas apresentam grande flexibilidade, possibilitando a modelagem de uma grande variedade de sistemas que apresentam contenção de recursos, como por exemplo, os sistemas de computação e redes de computadores e outros que podem ser modelados usando o paradifma de redes de filas.

Durante a descrição de um modelo, a interface verifica a consistência de cada elemento descrito individualmente e também valida o modelo globalmente, antes de submetê-lo a uma avaliação. A interface também oferece facilidades tais como manipulação de arquivos e acesso direto aos comandos do sistema operacional MS-DOS.

Após validar o modelo globalmente, a interface, que faz parte do ambiente *SAVAD*, procura uma solução para o modelo e fornece as medidas de desempenho solicitadas. Neste caso, o sistema deve escolher adequadamente as técnicas utilizadas, sejam elas exatas ou aproximadas. Mais detalhes sobre a interface do ambiente *SAVAD* podem ser obtidos em **[16]**.

3. O Processo de Modelagem usando *ALLOS*

3.1. Elementos de modelagem

O conjunto de elementos oferecidos pela ferramenta *ALLOS* permite a construção de modelos de redes de filas. Facilidades foram observadas na definição desses elementos para permitir modelar com flexibilidade sistemas de redes de computadores e de computação. A seguir, apresentamos um resumo dos elementos de modelagem. Mais detalhes podem ser obtidos em **[9,10]**.

Os elementos de modelagem utilizados são:

Clientes: são entidade temporárias que trafegam pelos nodos (entidades permanentes) do modelo requesitando serviços. O caminho seguido por um cliente é denominado ***rota***. Clientes com mesmos atributos (e.g. nível de prioridade) podem pertencer a uma mesma classe.

Estações de Trabalho: representam os recursos em um modelo de redes de filas.

Pontos de Controle: são nodos que controlam o fluxo de clientes em um modelo de rede de filas. Esses elementos foram projetados para facilitar a tarefa de construção de modelos de redes de computadores, particularmente aqueles que apresentam integração de serviços (e.g. voz e dados). Os pontos de controle são os seguintes:

— **Multiplicação**: permite a multiplicação de clientes que chegam a este nodo.

— **Ponto de Fusão**: faz a fusão de um ou mais clientes pertencentes a uma mesma classe, liberando um único cliente.

— **Ponto de Sincronização**: faz o bloqueio de clientes até que uma dada condição seja satisfeita.

— **Ponto Escalonador**: permite escalonar clientes conforme uma disciplina de liberação. As disciplinas de liberação disponíveis são as seguintes:

- **Cíclica:** clientes em fila são escalonados conforme ordem designada para cada fila.
- **Randômica:** escalona um cliente de uma fila randonicamente.
- **Livre:** uma fila pode tentar liberar seus clientes sem restrições. Contudo, colisões podem ocorrer quando duas ou mais filas tentam liberar clientes em um mesmo tempo. Na ocorrência de colisões, as iberações de clientes ocorrem conforme um algoritmo de reescalonamento.

Fonte: somente em redes abertas, representa o processo de chegada de clientes no modelo.

Sorvedouro: somente utilizado em redes abertas, eliminam clientes no modelo.

Classes: clientes com mesmos atributos. Clientes de mesma classe têm iguais prioridades.

3.2. Geração Automática de Estados

Modelos de sistemas reais normalmente dão origem a uma cadeia de Markov com um número elevado de estados. Um problema não menos importante diz respeito à geração automática de estados e taxas (ou probabilidades) de transição entre estados gerados **[1,2]**. É importante que o usuário possa especificar o modelo a partir de uma ferramenta amigável, e que as medidas de interesse possam também ser observadas de forma automática, sem que o usuário necessite conhecer os detalhes da representação matemática.

Em muitos modelos de sistemas de computação a maioria do tempo é gasto em um número relativamente pequeno de estados, em comparação ao número total de estados do modelo. Para tais modelos, várias medidas de interesse podem ser calculadas a partir de um número relativamente pequeno de estados **[2]**.

A idéia básica de técnicas de exploração dinâmica é a de desenvolver algoritmos para guiar o processo de geração da matriz de transição de estados. A importância de um estado deve ser dada em termos da sua contribuição para o cálculo de medidas de interesse, como por exemplo, fator de utilização e tempo médio de resposta de estações de serviço, vazão, tempo médio de espera em fila, comprimento médio de fila e outras **[2,17]**.

Uma forma elegante e eficiente para exploração dinâmica é a geração automática de estados, onde a partir da especificação do modelo de um sistema por parte do usuário, a matriz

de transição de estados é gerada a partir de um estado inicial e das regras que descrevem o comportamento do sistema.

Diversas ferramentas têm sido desenvolvidas para analisar o desempenho de sistemas de computação **[18,19,20,21,22]**. Grande parte dessas ferramentas usam a modelagem markoviana como modelo matemático que descreve o comportamento do sistema sendo analisado. Em **[1]** foi proposto uma ferramenta que implementa o paradigma de modelagem orientado a objeto, permitindo a descrição de modelos em alto nível. Seu domínio de aplicação abrange análise de desempenho e confiabilidade de sistemas. Uma breve comparação desta ferramenta com *ALLOS* mostra que *ALLOS* restringe seu domínio de aplicação para a modelagem e avaliação de desempenho de sistemas, onde supõem-se que estes são perfeitamente confiáveis. Entretanto, ALLOS possui certas vantagens, no sentido de solucionar os modelos markovianos, apresentar medidas de desempenho, além de estar disponível para uso em ambiente MS-DOS e UNIX (workstation). Ressaltamos que *ALLOS* também não exige do usuário qualquer conhecimento prévio em linguagens de programação (*PROLOG ou C++*), bem como na definição matemática de modelos.

A linguagem escolhida para a implemntação do módulo de geração automática de estados da ferramenta *ALLOS* foi *PROLOG*. Conforme discutido em **[1,3]**, são três os motivos principais desta escolha: *PROLOG* não exige a especificação dos tipos de dados utilizados, permitindo inteira liberdade na descrição dos estados dos modelos; *PROLOG* fornece unificação, uma forma poderosa de inicialização de variáveis e casamento de padrões usados na verificação das pré-condições; e, finalmente, *PROLOG* possui "backtrack": quando mais de uma regra tem suas pré-condições satisfeitas ao mesmo tempo, todas elas são testadas para achar todos os estados alcançados.

O decréscimo no custo de memória de computadores, o aumento da capacidade de processamento e o avanço das técnicas para solução de modelos, permitem que processos markovianos com um grande número de estados sejam, agora solucionados, reduzindo a forte restrição imposta pela modelagem markoviana, que é a explosão de estados **[2,3]**.

3.3. Procedimentos de geração automática de estados

Considere o esquema da Figura 1, que representa as possíveis transições de estados dos modelos que utilizam os elementos de modelagem apresentados na seção §2.1.

Na convenção adotada, cada elemento de modelagem pode ser precedido e seguido por uma ou mais filas. Exceções se aplicam aos elementos fonte (não pode ser precedido por fila) e sorvedouro (não pode ser seguido por fila). A notação usada, nesses casos, "[ ]", representa uma fila nula.

$$[\text{fila}_i] \left\{ \begin{array}{c} \text{fonte} \\ \text{sorvedouro} \\ \text{servidor} \\ \text{multiplicação} \\ \text{escalonador} \\ \text{sincronizador} \\ \text{fusão} \end{array} \right\} [\text{fila}_j]$$

Figura 1: Elementos de modelagem.

A partir de um estado $(k_1,...,k_n)$, onde n é o número de filas do modelo, o conjunto de estados alcançáveis a partir desse estado é definido como:

1. [ ] fonte $[f_j]$

Se $k_{f_j} < Max_{f_j}$

Então $(k_{f_1},...,k_{f_j},...,k_{f_n}) \underset{1}{\Rightarrow} (k_{f_1},...,k_{f_j}+1,...,k_{f_n})$

2. $[f_i]$ sorvedouro [ ]

Se $k_{f_i} > 0$

Então $(k_{f_1},...,k_{f_i},...,k_{f_n}) \underset{2}{\Rightarrow} (k_{f_1},..., k_{f_i}-1,...,k_{f_n})$

3. $[f_i]$ servidor $[f_j]$

Se $(k_{f_i} > 0) \wedge (k_{f_j} < Max_{f_j})$

Então $(k_{f_1},...,k_{f_i},...,k_{f_j},...,k_{f_n}) \underset{3}{\Rightarrow} (k_{f_1},...,k_{f_i}-1,...,k_{f_j}+1,...,k_{f_n})$

4. $[f_i]$ multiplicação $[f_{j_1},...,f_{j_m}]$

Se $k_{f_j} > 0$

Então $(k_{f_1},...,k_{f_i},...,k_{f_{j_\alpha}},...,k_{f_n}) \underset{4}{\Rightarrow} (k_{f_1},...,k_{f_i}-1,...,k_{f_{j_\alpha}}+1,...,k_{f_n})$,

$\forall\alpha \mid 1 \le \alpha \le m,\ k_{f_{j_\alpha}} < Max_{f_{j_\alpha}}$

5. $[f_{i_1},...,f_{i_m}]$ escalonador $[f_j]$

$\forall\alpha \mid 1 \le \alpha \le m$

Se $(k_{f_{i_\alpha}} \ge PE\ f_{i_\alpha}) \wedge (k_{f_j} + PE\ f_{i_\alpha} \le Max_{f_j})$

Então $(k_{f_1},...,k_{f_{i_\alpha}},...,k_{f_j},...,k_{f_n}) \underset{5}{\Rightarrow} (k_{f_1},...,k_{f_{i_\alpha}}-1,...,k_{f_j}+1,...,k_{f_n})$

6. $[f_{i_1},...,f_{i_m}]$ sincronizador $[f_{j_1},...,f_{j_p}]$

Se $(\forall\alpha \mid 1 \le \alpha \le m,\ k_{f_{i_\alpha}} \ge PS1_{f_{i_\alpha}}) \wedge (\forall\beta \mid 1 \le \beta \le p,\ k_{f_{j_\beta}} + PS2_{f_{j_\beta}} \le Max_{f_{j_\beta}})$

Então $(k_{f_1},...,k_{f_{i_\alpha}},...,k_{f_{j_\beta}},...,k_{f_n}) \underset{6}{\Rightarrow} (k_{f_1},...,k_{f_{i_\alpha}} - PS1_{f_{i_\alpha}},...,k_{f_{j_\beta}} + PS2_{f_{j_\beta}},...,k_{f_n})$

7. $[f_{i_1},...,f_{i_m}]$ fusão $[f_j]$

Se $(\forall\alpha \mid 1 \le \alpha \le m,\ k_{f_{i_\alpha}} \ge 0) \wedge (k_{f_j} < Max_{f_j})$

Então $(k_{f_1},...,k_{f_{i_\alpha}},...,k_{f_j},...,k_{f_n}) \underset{7}{\Rightarrow} (k_{f_1},...,k_{f_{i_\alpha}}-1,...,k_{f_j}+1,...,k_{f_n}),\ \forall\alpha \mid 1 \le \alpha \le m$

onde:

k_i é o comprimento da fila i,

Max_i é o comprimento máximo da fila i,

PE_i é o parâmetro associado à liberação de clientes da filas i do ponto escalonador,

$PS1_i$ é o parâmetro associado à liberação de clientes da fila i do ponto de sincronização,

$PS2_j$ é o parâmetro associado à liberação de clientes do ponto de sincronização para a fila j,

e onde:

$E_a \overset{1}{\underset{i}{\Rightarrow}} E_b$, ou simplesmente $E_a \underset{i}{\Rightarrow} E_b$, representa a existência de uma transição direta, correspondente ao elemento de modelagem i, entre os estados E_a e E_b.

Sejam E_a e E_b dois estados. Diz-se que E_b é alcançável a partir de E_a ($E_a \overset{*}{\Rightarrow} E_b$) se e somente se existe um número finito de transições entre E_a e E_b ($E_a \Rightarrow E_{a+1} \Rightarrow \ldots \Rightarrow E_b$). A transição de estado E_a para um E_b, é dito ser *válida* ($E_a \overset{val}{\Rightarrow} E_b$) se e somente se $\nexists E_c \mid E_b \underset{i}{\Rightarrow} E_c$, $5 \leq i \leq 7$. Assim, a geração de uma transição válida é feita segundo a seguinte regra:

$$\textbf{\textit{Se}}\ \exists E_c \mid ((E_b \underset{i}{\overset{*}{\Rightarrow}} E_c, 5 \leq i \leq 7) \wedge \nexists E_d \mid ((E_c \underset{i}{\Rightarrow} E_d, 5 \leq i \leq 7))$$

$$\textbf{\textit{Então}}\ E_a \overset{val}{\Rightarrow} E_c$$

$$\textbf{\textit{Senão}}\ E_a \overset{val}{\Rightarrow} E_b, \text{ desde que } \nexists E_c \mid E_b \underset{i}{\overset{*}{\Rightarrow}} E_c, 5 \leq i \leq 7$$

Essa regra pode ser ainda enriquecida pela consideração de restrições. Por exemplo, a restrição $\sum_{i=1}^{n} k_i = população$, onde k representa o tamanho da fila i e n o número total de filas do modelo, define um modelo de rede de filas fechado.

De outra forma, dada uma rede de filas F, um estado inicial E_0 é definido genericamente como:

If F é aberta (contém elementos do tipo fonte)

Então $E_0 = (0, \ldots, 0)$

Senão $E_0 = (P, 0, \ldots, 0)$

ondé P é a população da cadeia fechada. Caso $P > K_1$, então o excedente passará para K_2, e assim sucessivamente enquanto $P < K_i$. Esse estado inicial pode também ser definido diferentemente na presença de restrições.

Finalmente, o procedimento de geração da cadeia de Markov é definido como:

A partir de um conjunto de estados formado de um único estado inicial E_0.

```
constroi-estado([E|Es]) :- If E ainda não foi contruído,
                              Então X = {Estado: E ⇒ Estado}
                                    constroi-estado(Es°X)
                              Senão constroi-estado(Es)
```

onde $A \circ B$ significa a concatenação entre as duas listas A e B.

4. Exemplo numérico

Consideremos o exemplo apresentado na Figura 2, representando o modelo do protocolo de sessão com quarentena de dados local e modo de diálogo duplex **[23,24]**. Nesse modelo, UDSSs chegam à entidade de sessão de referência e são quarentenados (como UDPSs) até a composição de uma Unidade de Quarentena de Dados (UQD), quando então podem ser liberadas e enviadas pela conexão de sessão à entidade de sessão par. UDPSs que chegam à entidade de sessão par, são entregues ao usuário SS receptor. Uma UDPS corresponde a uma UDSS.

A capacidade de armazenamento na entidade de sessão de referência é de 6 UDPSs. A taxa média de transmissão na conexão de transporte atribuída a esta conexão de sessão é de 10 UDSTs por segundo. A conexão de transporte tem capacidade de armazenamento de 6 UDSTs. Na Figura 2, a fonte (Fonte1) representa a geração de UDSSs, o ponto de sincronização (Pt_Sincr) representa o processo de formação e liberação de uma UQD, a estação de serviço (Serv) representa a conexão de transporte e o sorvedouro (Sorv) representa a entrega de UDSSs à entidade de sessão par.

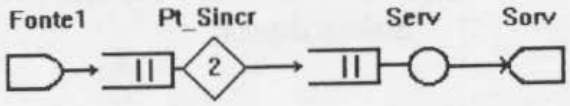

Figura 2: Modelo do Protocolo de Sessão com Quarentena de Dados e Modo de Diálogo Duplex.

Sumarizaremos, agora, a especificação que um usuário de ALLOS poderia dar a cada um dos elementos do modelo, através de sua interface:

Fonte:

Nome: Fonte1 (*UDSSs*)
Distribuição: exponencial
Media: 0,08333

Ponto de Sincronização:

Nome: Pt_Sincr (*UQD*)
Número de filas: 1
Comprimento de fila: 6
Associação: 2

Estação de Serviço:

Nome: Serv (*conexão de transporte*)
Comprimento de fila: 6
Tipo de servidor: simples
Distribuição: exponencial
Média: 0,1

Sorvedouro:

Nome: Sorv (*entrega de UDSSs à entidade de sessão par*)

Classe:

Nome: Classe1
Prioridade: 0 (sem prioridade)

Rota:

Nome: Rota1
Classe: Classe1
Sequência de nodos: Fonte1 >> Pt_Sincr.1 >> Serv >> Sorv

O espaço de estados associado ao modelo em estudo é o mostrado na Figura 3 e as transições entre os estados, na Figura 4.

Espaço de estados:

[01] (0,0)	[02] (1,0)	[03] (0,2)	[04] (1,2)	[05] (0,1)	[06] (1,1)
[07] (0,4)	[08] (0,3)	[09] (1,4)	[10] (1,3)	[11] (0,6)	[12] (0,5)
[13] (1,6)	[14] (1,5)	[15] (2,6)	[16] (2,5)	[17] (3,6)	[18] (3,5)
[19] (4,6)	[20] (4,5)	[21] (5,6)	[22] (5,5)	[23] (6,6)	[24] (6,5)

Figura 3: Espaço de estados para o modelo de protocolo de sessão com quarentena de dados e modo de diálogo duplex.

Transições entre os estados:

01 ⇒ 02:12	02 ⇒ 03:12	03 ⇒ 04:12	03 ⇒ 05:10	05 ⇒ 06:12	05 ⇒ 01:10	04 ⇒ 07:12	04 ⇒ 06:10
06 ⇒ 08:12	06 ⇒ 02:10	07 ⇒ 09:12	07 ⇒ 08:10	08 ⇒ 10:12	08 ⇒ 03:10	09 ⇒ 11:12	09 ⇒ 10:10
10 ⇒ 12:12	10 ⇒ 04:10	11 ⇒ 13:12	11 ⇒ 12:10	12 ⇒ 14:12	12 ⇒ 07:10	13 ⇒ 15:12	13 ⇒ 14:10
14 ⇒ 16:12	14 ⇒ 09:10	15 ⇒ 17:12	15 ⇒ 16:10	16 ⇒ 18:12	16 ⇒ 11:10	17 ⇒ 19:12	17 ⇒ 18:10
18 ⇒ 20:12	18 ⇒ 13:10	19 ⇒ 21:12	19 ⇒ 20:10	20 ⇒ 22:12	20 ⇒ 15:10	21 ⇒ 23:12	21 ⇒ 22:10
		22 ⇒ 24:12	22 ⇒ 17:10	23 ⇒ 24:10	24 ⇒ 19:10		

Figura 4: Transições entre os estados para o modelo de protocolo de sessão com quarentena de dados e modo de diálogo duplex.

· Para solucionar o modelo usando a ferramenta ALLOS, o usuário pode selecionar a opção solução "cadeias de Markov" disponível no menu principal da interface, ou, simplesmente deixar o SAVAD escolher esta solução, uma vez que este sistema pode escolher a técnica de solução mais adequada para um modelo, dando prioridade às técnicas analíticas.

A partir da geração automática do espaço de estados que comporá a matriz de transição do modelo, a ferramenta ALLOS soluciona o modelo usando o método apresentado em **[5]**. Outros métodos para a solução da matriz de transição de estados poderão ser encontrados em **[2]**.

Como exemplo de medidas de desempenho que podem ser obtidas, temos tempo médio de espera da fila do ponto de sincronização (atraso médio de admissão mais atraso médio de Quarentena de Dados) e atraso médio fia-a-fim.

- Tempo médio de espera em Pt_Sincr: 0,3
- Atraso médio fim-a-fim: 0,75

Os resultados acima foram comparados com **[23,24]**, com o objetivo de validar a ferramenta *ALLOS*, que é parte integrante do sistema SAVAD.

Ressaltamos que a validação desta ferramenta foi obtida através de estudos comparados exaustivamente com o Simulador de redes de filas, apresentado em **[24]**.

5. Conclusões

Do ponto de vista do usuário, é desejável que, em uma ferramenta de modelagem, tanto a definição matemática do sistema modelado quanto os detalhes referentes às técnicas de solução sejam transparentes, de forma que o usuário não precise ser um especialista nessas técnicas. Nesse sentido, a ferramenta *ALLOS*, com uma interface amigável, pode ser facilmente utilizada, não exigindo que seus usuários necessitem de conhecimentos mais aprofundados em avaliação de desempenho de sistemas, no que diz respeito à solução de modelos de redes de filas através do uso de cadeias de Markov.

ALLOS faz parte de um ambiente de simulação inteligente, denominado SAVAD. Esta ferramenta gera automaticamente os estados de modelos markovianos, solucionando-os, e apresenta as medidas de desempenho mais relevantes, solicitadas pelo usuário. A experiência mostrou que esta metodologia é simples, porém bastante eficiente. A ferramenta foi validada através da execução de testes exaustivos para verificar sua coerência com outras já existentes **[23,24]**.

A interface de *ALLOS* é bastante simples e amigável, oferecendo ao usuário facilidades para descrição de modelos em diversas aplicações. Ela é útil tanto na descrição de modelos e na validação destes, quanto na apresentação das medidas de desempenho obtidas após a solução dos modelos em análise.

A escolha das linguagens *PROLOG* e *C++* permitiram maior flexibilidade na implementação desta ferramenta, tornando o seu software modular e reutilizável, de forma a permitir facilmente extensões que venham a enriquecer as suas opções de utilização.

A ferramenta *ALLOS* foi projetada para funcionamento em equipamentos de pequeno porte, como micro-computadores compatíveis com a linha IBM-PC, tornando possível a sua utilização em larga escala. Além disso, pretende-se estender sua versatilidade para uso em estações de trabalho, em ambiente Unix.

Referências bibliográficas

[1] DINIZ, M. C., SILVA, E. de S. e "Especificação e Geração de Modelos Markovianos para Análise de Desempenho e Confiabilidade de Sistemas", *Revista Brasileira de Computação*, Vol. 6, n. 3, pp. 23-42, janeiro-março 1991.

[2] SILVA, E. de S. e, MUNTZ, R. R. "Métodos Computacionais de Solução de Cadeias de Markov: Aplicações a Sistemas de Computação e Comunicação", *VIII Escola de Computação*, Gramado (RS), 1992.

[3] SILVA, E. de S. e, "An object Oriented Methodology for the Specification of Markov Chains", *Relatórios Técnicos do NCE/UFRJ*, Rio de Janeiro (RJ), abril, 1988.

[4] KLEINROCK, L. "Queueing Systems", Vol. 1: Theory, Wiley Interscience, New York (USA), 1985.

[5] SAUER C. H., CHANDY, K. M. "Computer Systems Performance Modeling", Prentice-Hall, Englewood Cliffs (USA), 1981.

[6] KOBAYASHI, H. "Modeling and Analysis: An Introduction to System Performance Evaluation Methodology", Addison-Wesley, 1978.

[7] SHRIBER, T. J. "Simulation Using GPSS", John Willey & Sons, 1974.

[8] BIRTWISTLE, G. M. et al. "Simula Begin", Auerbach Publisher, Philadelphia (USA), 1973.

[9] CABRAL, M. I. C.; SOUTO, F. A. C.; CASTRO FILHO, H. C.; SILVA, H. DE M.; BRASILEIRO, M. A. G., SILVA, H. M. "An integrated system for modeling and evaluating models of networks of queues", in M. H. HAMZA (Editor), *Proceedings of the IASTED International Conference - Modeling and Simulation - MS'94 - Pittsburgh (USA)*, IASTED, Anaheim (USA), 1994.

[10] DIAS, M. M., BRASILEIRO, M. A. G., CABRAL M. I. C. "An Expert System for Performance Evaluation of Computer System Models", *Summer Computer Simulation Conference*, Reno (USA), 1992.

[11] SAUER, C. H., MACNAIR, E. "The Evaluation of the Research Queueing Package RESQ", in *Modelling techniques and Tools for Performance Analysis*, North-Holland, pp. 5-24, 1985.

[12] MCCOMAS, M. G., LAW, A. M. "Simulation Software for Communications Networks: The State of the Art", IEEE Comm. Magazine, March, 1994, pp. 44-50.

[13] BOOCH, G. "Object Oriented Design", The Benjamim/Cummings Publishing, 1991.

[14] STROUSTRUP, B. "The C++ Programming Language", Addison-Wesley, 1991.

[15] CLOCKSIN, W. F., MELLISH, C. S. "Programming in Prolog", Springer-Verlag, 1981.

[16] SOUTO, F. de A. C. "SAVAD - Sistema de Avaliação de Desempenho de Modelos de Redes de Filas". *Tese de Mestrado*, COPIN/CCT/UFPB, 1993.

[17] SILVA, E. de S. e; OCHOA, P. M. "State Space Exploration in Markov Models", *Performance Evaluation Review*, Vol. 20, n.1, p152-166, junho, 1992.

[18] MAKAN, S. V.; AVIZIENIS, A. ARIES 81: a rliability and life-cycle evaluation tool. *Proceedings of FTCS-12*, p274-276, 1982.

[19] CHIOLA, G. "A software package fo the analysis of generalized stochasric Petri net models". *Proceedings of the International Workshop on Timed Petri-nets*, p136-143, 1985.

[20] GOYAL, A.; CARTER, W.C.; SILVA, E.de S. e; LAVENBERG, S.S.; TRIVEDI, K.S. "The system availability estimator". *Proceedings of FTCS-16*, p84-89, 1986.

[21] JOHNSON Jr., A.M.; MALEK, M. "Survey of software tools for evaluating reliability, availability and serviceability". *ACM Computing Surveys*, 1988, 20:227-271.

[22] CIARDO, G. ; MUPPALA, J.; TRIVEDI, K. S. SPNP: Stochastic Petri net package. *Proceedings of the Third International Workshop on Petri-nets and Performance Modelos*, p142-151, 1989.

[23] CABRAL, M. I. C. "Modelagem do Protocolo da Camada de Sessão". *Tese de Doutorado*, CCPgEE/CCT/ UFPB, 1987.

[24] CONCEIÇÃO FILHO, H. C. "SIM/SAVAD - Um Simulador de Modelos de Redes de Filas". *Tese de Mestrado*, COPIN/CCT/UFPB, 1993.