Modelagem de Sistemas de Software com Redes de Petri Estocásticas

Guy Barroso
Depto. de Estatística e Computação
Universidade Estadual do Ceará
60715 Fortaleza, Ceará
Telefone: (085) 245-1133

Virgílio Almeida
Depto. de Ciência da Computação
Universidade Federal de Minas Gerais
30161 Belo Horizonte, Minas Gerais
Telefone: (031) 443-4088

Agosto 1990

Resumo

As redes de Petri vêm sendo cada vez mais utilizadas como um modelo adequado para a representação de sistemas de computação. As redes de Petri estocásticas aliam à capacidade descritiva das redes de Petri convencionais, a noção de tempo, permitindo, portanto, o cálculo e a análise de medidas de desempenho dos sistemas modelados.

Este artigo descreve a ferramenta FARPE, desenvolvida para a solução automática de redes de Petri estocásticas (i.e., a computação da distribuição das probabilidades estacionárias dos estados alcançados pelo modelo), e mostra a aplicação dessa ferramenta à modelagem e análise de construções do tipo *fork/join*, amplamente encontradas em sistemas de processamento paralelo e distribuído.

1 Introdução

As redes de Petri vêm sendo cada vez mais utilizadas como um modelo adequado para a representação de sistemas de computação. Basicamente, elas oferecem várias características positivas para a modelagem de sistemas. Além da capacidade descritiva, as redes de Petri permitem a análise do comportamento do sistema modelado, através de sua dinâmica de funcionamento. Como os modelos de redes de Petri podem ser construídos usando enfoques *bottom-up* e *top-down*, é possível usar sistematicamente esses modelos como ferramenta para projetar sistemas e verificar antecipadamente o seu comportamento.

Este trabalho apresenta a aplicação de uma classe especial de redes de Petri, as redes estocásticas, à modelagem de sistemas de software. O modelo escolhido é aquele determinado pelas construções *fork/join*, freqüentemente encontradas em sistemas de processamento paralelo e distribuído. O modelo é analisado usando a ferramenta FARPE, desenvolvida com a finalidade de automatizar a solução das redes de Petri estocásticas (SPN) e generalizadas (GSPN).

A organização do artigo é a seguinte: a segunda seção aborda aspectos da modelagem de sistemas de software. A terceira seção introduz os conceitos da teoria das redes de Petri e apresenta os modelos de redes estocásticas que serão tratados. A quarta seção apresenta o sistema FARPE como ferramenta de modelagem e análise de SPN e GSPN. A quinta seção mostra um exemplo real de modelagem e análise através da FARPE. A conclusão (sexta seção) apresenta um balanço do traballho e vantagens do sistema FARPE. No apêndice mostra-se a listagem de saída da FARPE para o exemplo da quinta seção.

2 O Problema da Modelagem de Sistemas

A maior parte das técnicas de especificação de sistemas descritas na literatura são orientadas a dados. Dentre essas técnicas estão a *Structured Analysis and Design Technique* (SADT) [21], a *Problem Statement Language and Problem Statement Analyser* (PSL/PSA) [23], a *Structured System Analysis* (SSA) [9, 10], e a *Hierarchy-Input-Process-Output* (HIPO) [22]. Provavelmente, a técnica mais compreensível para sistemas orientados ao controle seja a *Software Requirements Engineering Methodology* (SREM), que inclui a *Requirements Statement Language* (RSL) [1]. Entretanto, várias dessas metodologias não permitem especificações de tempo, e, quando permitem, fazem-no de maneira informal.

Problemas relacionados a sistemas como os de tempo real têm causado o desenvolvimento de aproximações através de extensões feitas a técnicas existentes, como os tipos abstratos de dados [6], a lógica temporal [5], as técnicas das asserções [11], e as máquinas de estado finito [8]. Em geral estas aproximações não consistem em mecanismos formais para modelar, por exemplo, um sistema de tempo real na abordagem *top-down*.

As redes de Petri têm sido vistas como uma ferramenta de modelagem adequada a essa classe de sistemas, por possuirem características como formalidade, capacidade de representar concorrência e sincronização, e adaptabilidade à decomposição hierárquica.

A modelagem de sistemas de software com redes de Petri apresenta vantagens em potencial. O sistema torna-se, em geral, mais inteligível, por causa da representação gráfica e da precisão apresentada. O comportamento dinâmico do sistema pode ser analisado usando-se a teoria das redes de Petri. Podem ser aplicadas técnicas para a verificação de programas paralelos [12, 13, 19]. Fica então possível projetar sistemas e saber a priori aspectos do comportamento e da correção desses sistemas. Além disso há a possibilidade de tratar as redes de Petri segundo os enfoques *bottom-up* e *top-down*.

3 Aplicação da Teoria das Redes de Petri

3.1 Conceitos Fundamentais sobre Redes de Petri

Basicamente, uma rede de Petri $RP = (P, T, A, M_0)$ é um modelo gráfico composto de lugares (P), transições (T), arcos (A), e uma marcação inicial (M_0). Sua estrutura é

formalmente especificada da seguinte maneira:

$$
\begin{aligned}
RP &= (P, T, A, M_0), \\
P &= \{p_1, p_2, p_3, \ldots, p_n\}, \\
T &= \{t_1, t_2, t_3, \ldots, t_m\}, \\
I &\subseteq (P \times T), \\
O &\subseteq (T \times P), \\
A &= (I \cup O), \\
M_0 &= \{M_0(p_1), M_0(p_2), \ldots, M_0(p_n)\}
\end{aligned}
$$

As redes possuem propriedades dinâmicas resultantes de sua execução, que é controlada pela posição e movimento dos *tokens* existentes na rede. É dito que a rede de Petri *executa* ao disparar transições. Uma transição é habilitada a disparar quando cada um dos seus lugares de entrada contém pelo menos tantos *tokens* quantos forem os arcos de entrada existentes ligando o lugar à transição. Dada uma marcação (i.e., a posição corrente dos *tokens*), o conjunto de todas as transições habilitadas é formado. Uma transição desse conjunto é selecionada para disparar, de acordo com algum modelo probabilístico ou em função de prioridades. Ao disparar, a transição selecionada remove *tokens* dos seus lugares de entrada e adiciona *tokens* nos seus lugares de saída (um *token* para cada arco existente).

O estado de uma rede de Petri é definido por sua marcação. Uma mudança de estado é causada pelo disparo de uma transição. Para uma rede com marcação M, a marcação M' é dita alcançável a partir de M se existe uma seqüência de transições que cause a mudança do estado M para o estado M'. Com isto se define o conjunto de alcance para uma rede como sendo o conjunto de todas as marcações alcançáveis a partir do estado inicial M_0.

3.2 Enfoques Básicos

Apresenta-se, a seguir, dois enfoques para a aplicação prática das redes de Petri ao projeto e análise de sistemas de software [20].

O primeiro enfoque utiliza técnicas convencionais de projeto para realizar a especificação do sistema. O sistema é modelado como uma rede de Petri (RP), e o modelo de RP é analisado. Problemas encontrados na análise da RP indicam falhas do projeto que deverá então ser modificado para corrigí-las. O projeto modificado será então modelado e analisado novamente. Este ciclo repete-se até que não sejam verificados problemas inaceitáveis na análise da RP. Na Figura 1 apresenta-se um diagrama desse processo. Cabe ressaltar que essa aproximação também pode ser usada para analisar um sistema já existente e operacional.

Observa-se que o primeiro enfoque considera a RP como ferramenta auxiliar de análise. Já o outro enfoque faz uso das redes de Petri para a especificação e projeto do sistema. As técnicas de análise das redes são utilizadas somente o necessário para garantir a ausência de erros nos modelos de RP.

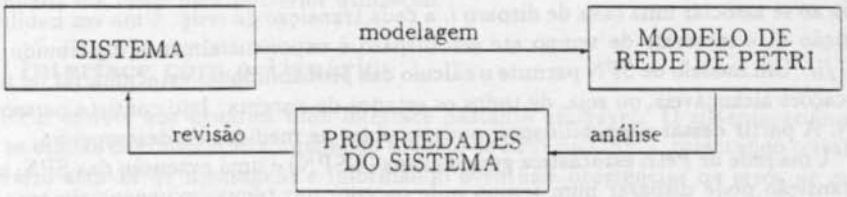

Figura 1: Modelagem de sistemas com redes de Petri.

A principal dificuldade do segundo caso consiste em desenvolver técnicas de implementação para transformar uma representação de *RP* num sistema executável. No primeiro caso, o problema é o desenvolvimento de técnicas de modelagem para transformar sistemas em representações de *RP*. Em ambos os casos, o interesse concentra-se no estudo das propriedades apresentadas pelas redes de Petri, que constituem o modelo do projeto.

3.3 Motivação para a Inclusão da Noção de Tempo nas *RP*

Correção e desempenho são aspectos cruciais a serem considerados no projeto de sistemas. Muito esforço tem sido dedicado à modelagem de sistemas e prova de correção. Utiliza-se tipicamente o cálculo de predicados ou grafos de precedência [17]. O interesse é a obtenção de um modelo que além de possuir propriedades formais verificáveis, seja também facilmente inteligível.

Embora o modelo original das redes de Petri tenha sido concebido para representar sistemas, não existe uma medida de tempo inerente ao modelo. Essa característica torna as redes de Petri originais inadequadas para analisar o desempenho de sistemas de computação.

Para a análise do desempenho, modelos estocásticos baseados nas redes de filas são utilizados [2]. Em virtude de sua fundamentação matemática, os projetistas oferecem uma resistência natural para sua utilização.

Existe a tendência para o uso de um modelo integrado, que considere tanto o aspecto de correção do projeto, quanto o aspecto de seu desempenho. Essa integração se dá através de um modelo gráfico, que seja fácil de entender, e cuja estrutura possa ser verificada. Adiciona-se a esse modelo o conceito de operação estocástica através da temporização de suas operações, tornando assim possível a extração das medidas de desempenho do sistema modelado. O modelo estocástico de *RP* abordado neste trabalho é a rede de Petri estocástica generalizada (GSPN).

3.4 As Redes de Petri Estocásticas

Uma rede de Petri estocástica (SPN) é uma extensão da rede de Petri original. Uma SPN é formada ao se associar uma taxa de disparo l_i a cada transição da rede. Uma vez habilitada a transição t_i, o intervalo de tempo até seu disparo é exponencialmente distribuído com média $1/l_i$. Um modelo de SPN permite o cálculo das probabilidades estacionárias de todas as marcações alcançáveis, ou seja, de todos os estados do sistema. Isto consiste na solução da SPN. A partir dessas probabilidades, pode-se obter as medidas de desempenho.

Uma rede de Petri estocástica generalizada (GSPN) é uma extensão das SPN, onde cada transição pode disparar num tempo nulo ou com um tempo exponencialmente distribuído. Dessa forma, as transições são divididas em dois conjuntos: transições imediatas (i.e. tempo zero) e transições temporizadas.

4 A FARPE como Ferramenta de Modelagem

Esta seção descreve uma ferramenta de software que resolve automaticamente modelos de redes de Petri estocásticas (SPN) e generalizadas (GSPN) [3]. A partir de uma linguagem de especificação, o usuário do sistema FARPE descreve a estrutura da rede e sua inicialização. O sistema então gera o conjunto de estados que podem ser alcançados com o funcionamento da rede, e computa a probabilidade estacionária de cada um desses estados. A partir destas probabilidades, o usuário pode encontrar as medidas de desempenho do sistema modelado pela rede.

4.1 Descrição Geral do Sistema

O sistema FARPE foi desenvolvido na linguagem Pascal, e se encontra disponível para execução em micro-computadores da linha PC sob o sistema operacional IBM/DOS.

O sistema compõe-se de dois módulos principais: o EDITOR, que permite a carga de uma rede, ou seja, permite a geração da representação interna da GSPN ou SPN a partir da descrição feita pelo usuário, e o SOLUCIONADOR, que usa essa representação interna para verificar a consistência da rede, gerar o seu conjunto de alcance, e calcular a distribuição das probabilidades estacionárias dos estados alcançáveis pela rede.

O módulo SOLUCIONADOR, na sua função de cálculo das probabilidades estacionárias dos estados alcançáveis, implementa um processo de solução computacionalmente eficiente [16]. Ele utiliza as rotinas presentes no sistema ESPARSOS, que reúne as funções de manipulação de vetores e matrizes esparsas, e ainda implementa o método de eliminação de Gauss com pivoteamento escalonado [7] para a resolução de sistemas de equações lineares.

O módulo EDITOR oferece, dentre outras opções, a função de criação de uma rede de Petri. Esta função pode ser realizada interativamente com o usuário através de perguntas apresentadas na tela, ou, caso o modelo seja uma SPN, através do desenho da rede na tela (usando o módulo EDITOR GRÁFICO de SPN). A definição do modelo pode ainda ser feita através do arquivo de descrição da rede de Petri (DRP), que é um

arquivo-texto convencional com sintaxe própria para a especificação do modelo de *RP* [3]. Indiferentemente à forma de criação da rede, sua estrutura pode ser visualizada e/ou armazenada em disco para posterior utilização.

4.2 Interface com o Usuário

A FARPE oferece aos usuários uma interface bastante amigável. O sistema acompanha todas as etapas de construção e análise do modelo de SPN ou GSPN, orientando o trabalho do usuário através de mensagens e informando eventuais ocorrências de erros de análise e/ou especificação da rede.

A interface é implementada no módulo EDITOR DE TELA, que gerencia o menu principal e permite a comunicação dos demais módulos com o usuário. O EDITOR DE TELA faz uso do sistema JANELAS, que controla a apresentação de janelas sobrepostas na tela, fornecendo as funções de abertura/fechamento de uma janela, emissão de mensagens e avisos, posicionamento do cursor e escrita na última janela aberta, dentre outras.

4.3 A Saída do Sistema FARPE

O usuário pode selecionar a saída a ser produzida pela análise da rede de Petri, bastando para isso ativar a opção OPÇÕES/ESPECIFICAR A SAÍDA. Utilizando o menu mostrado na Figura 2, o usuário define o conteúdo do arquivo de saída (que é um arquivo texto convencional) e indica se, após a solução da rede, esse arquivo deve ser mostrado na tela ou enviado à impressora do sistema.

5 Modelagem de Concorrência/Sincronização com o Uso da FARPE

O valor de qualquer técnica de modelagem é constatado pela sua utilidade na solução de problemas práticos. Nesta seção, apresenta-se a aplicação de redes de Petri à análise dos mecanismos de *fork* e *join*, que se encontram presentes em várias áreas da computação [2]. O modelo é construído e analisado no sistema FARPE.

5.1 O Modelo de Concorrência/Sincronização

Neste modelo existem N processos independentes, onde cada um realiza uma série de iterações idênticas. Cada iteração é constituída por um período de processamento seqüencial, seguido de um período de processamento concorrente. Ao entrar no período de processamento concorrente, o processo executa uma operação primitiva (e.g., *fork*, *cobegin*, *create_task*), dando origem a K sub-tarefas idênticas, cada uma designada a um dos K processadores operando independentemente. Após se completarem as K sub-tarefas, elas são sincronizadas por outra operação primitiva (e.g., *join*, *coend*, *merge_task*), e o processo

Figura 2: Menu de seleção da saída do sistema FARPE.

entra no seu próximo período de processamento seqüencial, durante o qual ele é processado por um único processador.

A representação deste modelo através de uma GSPN é mostrada na Figura 3 e sua interpretação na Tabela 1. As transições t_1, t_2, e t_3 são temporizadas, representando os períodos de processamento. A transição t_4 é imediata, representando a sincronização das sub-tarefas, para que seja iniciado um novo período de processamento seqüencial. Inicialmente existem N *tokens* no lugar p_1 da rede, representando os N processos no sistema.

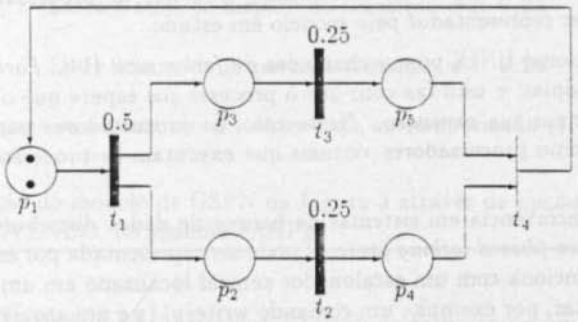

Figura 3: Representação de GSPN para o modelo de concorrência/sincronização.

LUGAR	DESCRIÇÃO
p_1	Processos em período de processamento seqüencial
p_2, p_3	Sub-tarefas em processamento paralelo
p_4, p_5	Sub-tarefas aguardando sincronização
TRANSIÇÃO	**DESCRIÇÃO**
t_1	Fim do período de processamento seqüencial
t_2, t_3	Fim do processamento da sub-tarefa
t_4	Sincronização das sub-tarefas e início do período de processamento seqüencial

Tabela 1: Legenda para a GSPN da Figura 3.

5.2 Algumas Interpretações para o Modelo

O modelo apresentado na seção anterior descreve naturalmente várias aplicações. Dentre elas estão as seguintes:

1. Um sistema *Cray Y-MP* com K processadores [17]. Esses sistemas suportam multitarefas para tentar diminuir o tempo de execução dos programas. Características especiais de *hardware* permitem que vários processadores sejam utilizados por um processo. O sistema operacional *Cray* utiliza tarefas como a unidade básica de trabalho. Outros sistemas multiprocessadores, tais como o *Sequent Balance 8000*, *Alliant FX*, e *Encore Multimax* suportam mecanismos similares de processamento concorrente, podendo ser representados pelo modelo em estudo.
2. O sistema operacional UNIX possui chamadas de *fork* e *wait* [14]. *Fork* divide o programa em duas cópias, e *wait* faz com que o processo pai espere que o filho termine, para então continuar sua execução. Neste caso, os processadores paralelos do modelo são vistos como processadores virtuais que executam os processos criados pela operação *fork*.
3. O controle de concorrência em sistemas de bancos de dados distribuídos. A versão centralizada do *two-phased locking protocol* pode ser representada por esse modelo [4]. Este protocolo funciona com um escalonador central localizado em um dos nodos da rede. Para executar, por exemplo, um comando write(y) (y é um arquivo armazenado em um dos nodos da rede), o programa tem que pedir um bloqueio para gravação em y ao escalonador central que, por sua vez, emite mensagens a todos os nodos. O escalonador central fica aguardando a confirmação de todos os nodos da rede de que o bloqueio foi feito, para então executar o comando write(y).

5.3 Construção do Modelo Através da FARPE

Para efeito de exemplificação, o modelo de concorrência/sincronização será analisado com os seguintes parâmetros: $N = 2$, $K = 2$, *tempo de processamento seqüencial* $= 2$, e *tempo de processamento paralelo* $= 4$. Assume-se, por exemplo, a segunda interpretação da seção anterior, representada pela GSPN da Figura 3 e Tabela 1.

Ao ativar a opção EDITAR/NOVA/DESCREVER A REDE, a FARPE apresenta uma seqüência de perguntas que são respondidas na ordem mostrada na Figura 4.

5.4 Análise do Modelo

A GSPN do modelo estudado foi descrita pela FARPE, que gerou a sua representação interna. A solução da rede é obtida ativando-se a opção RESOLVER. Assumindo a especificação de saída mostrada na Figura 2, a FARPE produz os resultados mostrados no apêndice.

Como ilustração do tipo de medida de desempenho que pode ser obtida com a FARPE, mostra-se a seguir o cálculo do tempo médio de processamento dos processos.

Número de lugares da rede: **5**
Número de transições da rede: **4**
Lugares de entrada da transição #1: **1**
Lugares de saída da transição #1: **2 3**
Informe se a transição #1 é imediata ou temporizada (I/T): **T**
Entre a taxa de disparo da transição #1: **0.5**
Lugares de entrada da transição #2: **2**
...
Entre a taxa de disparo da transição #2: **0.25**
...
Entre a taxa de disparo da transição #3: **0.25**
...
Informe se a transição #4 é imediata ou temporizada (I/T): **I**

Figura 4: Descrição do modelo de GSPN da Figura 3 através da opção EDITAR/NOVA/CRIAR DESCREVENDO, no sistema FARPE.

A lei de Little [15] estabelece que o tempo de resposta de um sistema é igual ao produto da taxa de processamento desse sistema pelo número médio de processos presentes. A taxa de processamento do sistema será igual ao fluxo de processamento através da transição t_1, que é dado por $l_1 \times (Prob(M_1) + Prob(M_2) + Prob(M_4) + Prob(M_5))$, já que as marcações tangíveis 1, 2, 4, e 5, são as únicas que habilitam a transição t_1. Uma vez calculada a taxa de processamento, obtém-se o tempo médio de resposta através da lei de Little. Com os valores resultantes da análise da rede, dados no apêndice, tem-se que o tempo médio de resposta será 11.39 unidades de tempo.

6 Conclusão

O artigo apresentou uma introdução à teoria das redes de Petri, e mostrou a aplicação dessa teoria à modelagem de sistemas de software. O sistema FARPE, descrito no artigo, encapsula todo o processo de solução das redes de Petri, ou seja, a geração do espaço de estados, validação desse espaço, e resolução do processo estocástico representado.

O sistema FARPE, desenvolvido para ser uma ferramenta efetiva de projeto e análise de sistemas, utiliza um processo computacionalmente eficiente [16], que permite que sejam modeladas características de sistemas reais. Além do encapsulamento mencionado, a FARPE oferece aos usuários uma interface bastante amigável.

Com o intuito de mostrar a utilidade da ferramenta, foi apresentado um modelo de rede de Petri da construção *fork/join*, amplamente utilizada em sistemas de processa-

mento paralelo e distribuído. Os procedimentos da FARPE necessários para a descrição do modelo, resolução da rede de Petri, e obtenção das medidas de desempenho foram também descritos.

Referências Bibliográficas

[1] Alford,M.W. "A requirements engineering methodology for real-time processing requirements", IEEE Trans. Software Eng., vol.SE-3, pp.60-69, Jan. 1977.

[2] Almeida,Virgílio and Dowdy,Lawrence "Performance Analysis of a Scheme for Concurrency/Synchronization Using Queueing Network Models". International Journal of Parallel Programming. Plenum Press, Vol.15, No.6, pp.529-550, December 1986.

[3] Barroso, Guy "FARPE - Uma Ferramenta para a Análise de Redes de Petri Estocásticas e Generalizadas". Dissertação de Mestrado, Depto. de Ciência da Computação, UFMG, 1990.

[4] Bernstein, B. and Goodman, N. "Concurrent Control in Distributed Database Systems", Computing Surveys, Vol.13, No.2, 1983.

[5] Bernstein, A. and Harter, P. K. "Proving real-time properties of programs with temporal logic", Proc. Eighth Symp. Operating Systems Principles, pp.1-11, Dec. 1981.

[6] Booth, T. L. and Wiecek, C. A. "Performance abstract data types as a tool in software performance analysis and design", IEEE Trans. Software Eng., vol.SE-6, pp.138-151, Mar. 1980.

[7] Cheney, E. and Kincald, D. "Numerical Mathematics and Computing", second edition, Brooks/Cole Publishing Company, Monterey, California, 1985.

[8] Dasarathy, B. "Timing constraints for real-time systems: constructs for expressing them, methods of validating them", Proc. IEEE Real-Time System Symp., pp.197-204, Dec. 1982.

[9] DeMarco, T. "Structured Analysis and System Specification", Englewood Cliffs, NJ, Prentice-Hall, 1979.

[10] Gane, C. "Data design in structured systems analisis", Infotech State of the Art Report on Data Design, 1980.

[11] Haase, V. H. "Real-time behavior of programs", IEEE Trans. Software Eng., vol.SE-7, pp.494-501, Sep. 1981.

[12] Hoare, C. A. R. "Parallel Programming, an Axiomatic Approach", Tech. Report C5-73-394, Stanford University, Oct. 1973.

[13] Keller, R. M. "Formal Verification of Parallel Programs", Comm. ACM, Vol.19, No.7, July 1976, pp.371-384.

[14] Kernigham, B. and Pike, R. "The UNIX Programming Environment", Prentice-Hall, 1984.

[15] Little, J. "A proof of the queueing formula $L = \lambda W$", Oper. Res. 9, pp.383-387, 1961.

[16] Marsan, A., Conte, G. and Balbo, G. "A Class of Generalized Stochastic Petri Net for the Performance Evaluation of Multiprocessor Systems", ACM Transactions on Computer Systems, Vol.2, No.2, May 1984.

[17] Molloy, M. "On the Integration of Delay and Throughput Measures in Distributed Processing Models", Ph.D. Thesis, University of California, Los Angeles, 1986.

[18] Myers, W. "Getting the Cycles of a Supercomputer", Computer, Vol.19, No.3, March 1986.

[19] Owicki, S. and Gries, D. "Verifying Properties of Parallel Programs: An Axiomatic Approach", Comm. ACM, Vol.19, No.5, May 1976, pp.279-285.

[20] Peterson, J. "Petri Net Theory and the Modelling of Systems", Prrentice-Hall, Englewood Clifs, N.J., 1981.

[21] Ross, D. T. "Structured Analysis (SA): A language for communicating ideas", IEEE Trans. Software Eng., vol.SE-3, pp.16-34, Jan. 1977.

[22] Stay, J. F. "HIPO and integrated system design", IBM Syst. J., vol.15, pp.143-154, 1976.

[23] Teichroew, D. and Hershey, E. A. "PSL/PSA: A computer-aided technique for structured documentation and analysis of information processing systems", IEEE Trans. Software Eng., vol.SE-3, pp.41-48, Jan. 1977.

Apêndice

```
QUANTIDADE DE LUGARES    DA REDE:   5
QUANTIDADE DE TRANSICOES DA REDE:   4

LUGARES DE ENTRADA E DE SAIDA DAS TRANSICOES   <LUGAR> ( <CARDINALIDADE> )
-------------------------------------------

TRANSICAO #    1: TEMPORIZADA
LUGARES DE ENTRADA:  1( 1)
LUGARES DE   SAIDA:  2( 1)  3( 1)
TRANSICAO #    2: TEMPORIZADA
LUGARES DE ENTRADA:  2( 1)
LUGARES DE   SAIDA:  4( 1)
TRANSICAO #    3: TEMPORIZADA
LUGARES DE ENTRADA:  3( 1)
LUGARES DE   SAIDA:  5( 1)
TRANSICAO #    4: IMEDIATA
LUGARES DE ENTRADA:  4( 1)  5( 1)
LUGARES DE   SAIDA:  1( 1)

MARCACAO INICIAL :   <LUGAR> ( <TOKENS> )
-----------------

 1( 2)

                              CONJUNTO DE ALCANCE
                              -------------------

EXISTEM 9 MARCACOES TANGIVEIS E 4 MARCACOES ESVAECENTES

MARCACOES TANGIVEIS
-------------------
Marcacao# 1:  1( 2)
Marcacao# 2:  1( 1)  2( 1)  3( 1)
Marcacao# 3:  2( 2)  3( 2)
Marcacao# 4:  1( 1)  3( 1)  4( 1)
Marcacao# 5:  1( 1)  2( 1)  5( 1)
Marcacao# 6:  2( 1)  3( 2)  4( 1)
Marcacao# 7:  2( 2)  3( 1)  5( 1)
Marcacao# 8:  3( 2)  4( 2)
Marcacao# 9:  2( 2)  5( 2)

MARCACOES ESVAECENTES
---------------------
Marcacao# 1:  1( 1)  4( 1)  5( 1)
Marcacao# 2:  2( 1)  3( 1)  4( 1)  5( 1)
Marcacao# 3:  3( 1)  4( 2)  5( 1)
Marcacao# 4:  2( 1)  4( 1)  5( 2)
```

```
TRANSICOES DE MARCACOES TANGIVEIS PARA MARCACOES TANGIVEIS
----------------------------------------------------------

(marcacao inicial) --> (marcacao final) ((transicao),(probabilidade),(taxa))

  1 --> 2  ( 1,  1.0000000000E+00,  5.0000000000E-01)
  2 --> 3  ( 1,  5.0000000000E-01,  5.0000000000E-01)
  2 --> 4  ( 2,  2.5000000000E-01,  2.5000000000E-01)
  2 --> 5  ( 3,  2.5000000000E-01,  2.5000000000E-01)
  3 --> 6  ( 2,  5.0000000000E-01,  2.5000000000E-01)
  3 --> 7  ( 3,  5.0000000000E-01,  2.5000000000E-01)
  4 --> 6  ( 1,  6.6666666667E-01,  5.0000000000E-01)
  5 --> 7  ( 1,  6.6666666667E-01,  5.0000000000E-01)
  6 --> 8  ( 2,  5.0000000000E-01,  2.5000000000E-01)
  7 --> 9  ( 3,  5.0000000000E-01,  2.5000000000E-01)

TRANSICOES DE MARCACOES TANGIVEIS PARA MARCACOES ESVAECENTES
-----------------------------------------------------------

(marcacao inicial) --> (marcacao final) ((transicao),(probabilidade),(taxa))

  4 --> 1  ( 3,  3.3333333333E-01,  2.5000000000E-01)
  5 --> 1  ( 2,  3.3333333333E-01,  2.5000000000E-01)
  6 --> 2  ( 3,  5.0000000000E-01,  2.5000000000E-01)
  7 --> 2  ( 2,  5.0000000000E-01,  2.5000000000E-01)
  8 --> 3  ( 3,  1.0000000000E+00,  2.5000000000E-01)
  9 --> 4  ( 2,  1.0000000000E+00,  2.5000000000E-01)

TRANSICOES DE MARCACOES ESVAECENTES PARA MARCACOES TANGIVEIS
-----------------------------------------------------------

(marcacao inicial) --> (marcacao final) ((transicao), (probabilidade))

  1 --> 1  ( 4,  1.0000000000E+00)
  2 --> 2  ( 4,  1.0000000000E+00)
  3 --> 4  ( 4,  1.0000000000E+00)
  4 --> 5  ( 4,  1.0000000000E+00)

TRANSICOES DE MARCACOES ESVAECENTES PARA MARCACOES ESVAECENTES
-------------------------------------------------------------

(marcacao inicial) --> (marcacao final) ((transicao), (probabilidade))

*** NAO EXISTEM.

              PROBABILIDADES ESTACIONARIAS DAS MARCACOES
              ------------------------------------------

Marcacao# 1:  8.1081081081E-02          Marcacao# 2:  1.0810810811E-01
Marcacao# 3:  1.0810810811E-01          Marcacao# 4:  8.1081081081E-02
Marcacao# 5:  8.1081081081E-02          Marcacao# 6:  1.3513513514E-01
Marcacao# 7:  1.3513513514E-01          Marcacao# 8:  1.3513513513E-01
Marcacao# 9:  1.3513513513E-01
```