Wagner Luiz Alves de Oliveira

Proposta de uma Rede de Petri de Alto - Nível Incluindo Temporização

cm 1 2 3 4 5 6 7 8 unesp 12 13 14 15 16 17 18 19

WAGNER LUIZ ALVES DE OLIVEIRA

PROPOSTA DE UMA REDE DE PETRI DE ALTO-NÍVEL INCLUINDO TEMPORIZAÇÃO

ORIENTADOR: NORIAN MARRANGHELLO

(MAC - 019)

Inst. de Biociências, Letras e Ciências Exatas - UNESP
BIBLIOTECA
S. J. Rio Preto
771/98

DISSERTAÇÃO APRESENTADA AO IBILCE - UNESP COMO PARTE DOS REQUISITOS EXIGIDOS PARA A OBTENÇÃO DO TÍTULO DE MESTRE EM MATEMÁTICA APLICADA E COMPUTACIONAL

SÃO JOSÉ DO RIO PRETO - 26 DE FEVEREIRO DE 1997

DEDICO ESTE TRABALHO AOS MEUS PAIS, OSMAR E CONCEIÇÃO,
MEUS IRMÃOS, SOLANGE E TADEU,
E À PESSOA QUE ESTARÁ SEMPRE AO MEU LADO, MARISA

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

AGRADEÇO À FAPESP PELO APOIO PRESTADO AO DESENVOLVIMENTO DO TRABALHO AQUI EXPOSTO, EXPRESSO ATRAVÉS DAS BOLSAS DE INICIAÇÃO CIENTÍFICA E DE MESTRADO

unesp

RESUMO

Redes de Petri constituem uma ferramenta matemática de modelagem de sistemas. Através de suas extensões, redes de Petri abrangem um grande campo de aplicações, tais como: controle de processos industriais, protocolos de comunicação, linguagens de programação, sistemas distribuídos etc. Tais modelos possibilitam a representação de sistemas com alto grau de paralelismo, os quais podem ser simulados nos níveis funcional, lógico e temporal com um grau razoável de simplicidade.

O presente trabalho tem como objetivos aprimorar e sistematizar uma extensão de rede de Petri, proposta no trabalho de iniciação científica do mestrando, bem como implementar um simulador computadorizado para ela.

A primeira parte da dissertação faz uma abordagem geral sobre redes de Petri e suas extensões mais conhecidas na literatura. São tratados os principais tipos das redes ordinárias e das redes de alto-nível. Além destas, são tratadas duas classes especiais de redes, as quais adicionam às redes de Petri importante força descritiva no modelamento e análise de propriedades temporais e de performance de sistemas.

A segunda parte da dissertação apresenta a proposta de uma extensão concebida a partir dos principais elementos funcionais e estruturais das redes de Petri Numéricas, Temporizadas e Coloridas, a qual foi denominada **Rede de Petri Temporizada de Alto-Nível** (RPTAN). Esta parte da dissertação apresenta, também, as características de implementação e funcionamento do ***Simulador RPTAN***, desenvolvido para a nova extensão de rede aqui proposta. Exemplos de modelamento em RPTAN são apresentados com o intuito de evidenciar a praticidade e potencialidade da ferramenta.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

ABSTRACT

Petri nets are a mathematical tool for system modeling. There are several Petri net extensions which achieve a wide number of applications, such as: industrial processes control, communications protocols, programming languages, distributed systems, and so on. These extensions enable the representation of highly parallel systems, which can be simulated at the temporal, logic, and functional levels with reasonable simplicity.

The goals of this work are to enhance and systematize a Petri net extension, proposed during the graduation work of the candidate, as well as to implement a computer simulator for it.

The first part of this dissertation has a general overview of Petri nets and its most well known ordinary and high-level extensions. Besides these, two other kinds of nets, which added descriptive power to the modeling and analysis of performance and temporal properties of systems, are discussed.

The second part of this dissertation proposes an extension conceived from the coloured, timed, and numerical Petri nets main structural and functional elements. This extension was called high-level timed Petri net (**RPTAN**, for *Rede de Petri Temporizada de Alto-Nível*). The characteristics of RPTAN simulator, developed to help in the analysis of models described in the proposed extension, are described. The analysis of two RPTAN models are presented to show the functionality and potentiality of this tool.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

ÍNDICE

INTRODUÇÃO 1

PARTE I: REDES DE PETRI E SUAS PRINCIPAIS EXTENSÕES

CAPÍTULO 1: FUNDAMENTOS BÁSICOS DE REDES DE PETRI
1.1. CRONOLOGIA 4
1.2. COMPOSIÇÃO BÁSICA DE UMA REDE DE PETRI 5
1.3. REDES DE PETRI ELEMENTARES 5
1.4. EXEMPLO DE MODELAGEM DE UMA BIBLIOTECA EM RPE 7

CAPÍTULO 2: REDES DE PETRI LUGAR / TRANSIÇÃO
2.1. REGRAS DE FUNCIONAMENTO 10
2.2. PROPRIEDADES GERAIS DE UMA RP LUGAR/TRANSIÇÃO
2.2.1. LIMITABILIDADE 14
2.2.2. VIVACIDADE 14
2.2.3. REINICIABILIDADE 16
2.3. EXEMPLO DE MODELAGEM DE UMA BIBLIOTECA EM RP L/T 17

CAPÍTULO 3: REDES DE PETRI TEMPORIZADAS
3.1. TIPOS DE REDES DE PETRI TEMPORIZADAS 18
3.2. RP TEMPORIZADAS COM VALORES DE TEMPO ASSOCIADOS ÀS TRANSIÇÕES 18
3.2.1. DESCRIÇÃO DAS RPT DE MERLIN 18
3.2.2. DESCRIÇÃO DAS RPT DE RAMCHANDANI 18
3.3. RP TEMPORIZADAS COM VALORES DE TEMPO ASSOCIADOS AOS LUGARES 19
3.4. RP TEMPORIZADAS DE MERLIN (RPTM) 19
3.5. MÉTODO DE ENUMERAÇÃO DE CLASSES DE ESTADOS PARA RPT DE MERLIN 23

CAPÍTULO 4: REDES DE PETRI NUMÉRICAS
4.1. ORIGEM DAS REDES DE PETRI NUMÉRICAS 28
4.2. REGRAS DE FUNCIONAMENTO 28
4.3. MODELO RPN DE UMA CÉLULA FLEXÍVEL DE MANUFATURA 31

CAPÍTULO 5: REDES DE PETRI ESTOCÁSTICAS
5.1. PRINCIPAIS CLASSES DE REDES DE PETRI ESTOCÁSTICAS 35
5.2. MUDANÇAS INTRODUZIDAS PELAS REDES DE PETRI ESTOCÁSTICAS
5.2.1. ASSOCIAÇÃO DE UM TEMPO DE DISPARO ALEATÓRIO ÀS TRANSIÇÕES 35
5.2.2. TRANSIÇÕES IMEDIATAS 36
5.2.3. ARCOS PROBABILÍSTICOS / CHAVES ALEATÓRIAS 36
5.2.4. ARCOS ANORMAIS 38
5.3. ESTIMATIVAS DE DESEMPENHO 39
5.4. PROCEDIMENTOS DE CÁLCULO DE ESTIMATIVAS DE DESEMPENHO 41

CAPÍTULO 6: REDES DE PETRI COLORIDAS
6.1. INTRODUÇÃO INFORMAL 44
6.2. DEFINIÇÃO FORMAL 50
6.3. PROPRIEDADES DINÂMICAS 55
6.4. GRAFO DE DISPAROS 58

unesp

PARTE II: REDES DE PETRI TEMPORIZADAS DE ALTO-NÍVEL

CAPÍTULO 7: DESENVOLVIMENTO TEÓRICO DE RPTAN

7.1. INTRODUÇÃO 64
7.2. REPRESENTAÇÃO E REGRAS DE FUNCIONAMENTO 65
7.3. DEFINIÇÃO DE RPTAN 70
7.4. EXEMPLO DE MODELAGEM EM RPTAN 74

CAPÍTULO 8: DESENVOLVIMENTO PRÁTICO DE RPTAN

8.1. INTRODUÇÃO 77
8.2. FUNCIONAMENTO DO SIMULADOR RPTAN 77
8.2.1. ENTRADA DOS LUGARES DA REDE 78
8.2.2. ENTRADA DAS TRANSIÇÕES DA REDE 79
8.2.3. ENTRADA DOS RAMOS DA REDE 84
8.2.4. SIMULAÇÃO DA REDE 90
8.3. CARACTERÍSTICAS DE IMPLEMENTAÇÃO DO SIMULADOR RPTAN
8.3.1. ESTRUTURAS DE DADOS 92
8.3.2. ALGORITMO DE SIMULAÇÃO 99
8.3.3. ALGORITMO DE VERIFICAÇÃO DA PROPRIEDADE DE REINICIABILIDADE 105
8.3.4. ALGORITMO DE VERIFICAÇÃO DA PROPRIEDADE DE VIVACIDADE 107

CAPÍTULO 9: EXEMPLOS DE MODELAGEM EM RPTAN

9.1. EXEMPLO DOS *DINING PHILOSOPHERS*
9.1.1. DESCRIÇÃO DO PROBLEMA 109
9.1.2. MODELO RPTAN DOS *DINING PHILOSOPHERS* 109
9.2. EXEMPLO DA ARQUITETURA ABACUS
9.2.1. DESCRIÇÃO DA ARQUITETURA ABACUS 114
9.2.2. MODELO RPTAN DA ARQUITETURA ABACUS 117

CONCLUSÕES 123

APÊNDICE: TÓPICOS DA TEORIA DE GRAFOS 124

REFERÊNCIAS BIBLIOGRÁFICAS 130

INTRODUÇÃO

Redes de Petri (RP) tratam-se de uma ferramenta matemática de modelagem de sistemas, criadas a partir da tese de doutoramento do pesquisador alemão Carl Adam Petri, intitulada "*Kommunication mit Automaten*".

Desde o princípio (década de 60), RP objetivaram a análise e a modelagem de sistemas com componentes concorrentes. Nas últimas décadas, a teoria de RP se desenvolveu, acarretando no aparecimento de extensões à mesma. Com tais extensões, RP passaram a abranger um campo maior de aplicações, atingindo áreas como:

- modelagem de componentes de *hardware*;
- controle de processos industriais;
- protocolos de comunicação;
- linguagens de programação;
- sistemas distribuídos;
- automação de escritórios;
- bancos de dados;
- inteligência artificial; e
- sistemas de informação.

Este trabalho de mestrado visa a caracterização teórica de uma ferramenta matemática de modelagem, baseada em RP e suas extensões de alto-nível. A motivação do mesmo nasceu da necessidade de simular, em um computador de pequeno porte, o comportamento dos componentes de uma arquitetura computacional dedicada à simulação de circuitos eletrônicos, altamente paralela e com fortes restrições temporais, proposta por N.Marranghello e F.Damiani (arquitetura ABACUS [24]). Para a construção do modelo da arquitetura em questão, inicialmente, adotou-se as Redes de Petri devido à sua potencialidade gráfica e facilidade de compreensão por terceiros. B.C.Damasceno et al. [7] demonstraram que uma RP clássica é pouco adequada no tratamento de questões temporais e de estruturas repetitivas, presentes no modelo da arquitetura ABACUS.

Assim, partiu-se para o estudo de diferentes extensões de RP, procurando encontrar dentre elas elementos que pudessem resolver os problemas de modelagem apresentados pela RP clássica. As RP Temporizadas de Merlin [5] mostraram-se adequadas para modelar os aspectos temporais da arquitetura ABACUS, enquanto as RP Numéricas [10] mostraram-se adequadas no tratamento das estruturas repetitivas. Entretanto, inexistia um simulador de RP para PC que unisse as características de ambas as redes. Desta forma, a partir das principais características operacionais das redes citadas, com a adição de outras características, formou-se a base de uma nova extensão de RP, a qual foi denominada Redes de Petri Numérico-Temporizadas (RPNT). A partir de um conjunto de definições iniciais, construiu-se um simulador para a nova extensão, o qual faz parte de um trabalho de iniciação científica (processo FAPESP 93/4986-2) [26,27].

Entretanto, durante o desenvolvimento de exemplos de modelos em RPNT, observou-se a necessidade de mudanças na caracterização dessa extensão, visando uma utilização mais fácil e direta no que dizia respeito ao processo de criação de um modelo de sistema. Assim, no presente trabalho de mestrado, fez-se um estudo mais



aprofundado dos principais tipos de RP, procurando subsídios que pudessem melhorar significativamente a representação e forma de funcionamento das RPNT. A partir deste estudo, encontrou-se nas RP Coloridas a representação mais completa e flexível dentre as diferentes extensões de RP. Desta forma, procurou-se transferir para as RPNT os princípios que caracterizam a potencialidade que as RP Coloridas têm em trabalhar sobre *tokens* diferenciáveis, gerando, assim, uma segunda versão de RPNT. Devido às suas características de RP de alto-nível, houve-se por bem denominar a segunda versão de RPNT por Rede de Petri Temporizada de Alto-Nível (RPTAN).

Este trabalho encontra-se dividido em duas partes. A primeira parte trata de Redes de Petri e suas principais extensões, enquanto a segunda parte trata das Redes de Petri Temporizadas de Alto-Nível.

As definições e conceitos básicos de RP são apresentados no primeiro capítulo. Inicialmente é dado o desenvolvimento cronológico da teoria de RP. Em seguida, por meio das RP Elementares, são apresentados a composição e conceitos básicos usados pelas principais extensões de RP.

O capítulo 2 apresenta as RP Lugar/Transição, com ênfase na formalização matemática do modelo. Tal formalização é importante, uma vez que as demais extensões de RP utilizam as RP Lugar/Transição como base.

As RP Numéricas são apresentadas no terceiro capítulo. Tais redes são uma generalização das RP originais, com a adição de grande força descritiva às mesmas.

O capítulo 4 traz três modelos de RP Temporizadas: as de Sifakis, de Ramchandani e de Merlin. A ênfase é dada para este último, uma vez que os modelos de Sifakis e de Ramchandani são considerados como um subconjunto do modelo de Merlin.

O capítulo 5 apresenta a extensão de RP utilizada para o cálculo de estimativas de desempenho, denominada Redes de Petri Estocásticas.

Encerrando a primeira parte, o capítulo 6 apresenta a extensão de RP mais utilizada nas aplicações práticas, as chamadas Redes de Petri Coloridas. Este modelo de rede é a principal representante das chamadas redes de alto-nível.

A segunda parte deste trabalho consiste dos capítulos de 7 a 9.

O capítulo 7 apresenta o desenvolvimento teórico, a representação, a forma de funcionamento e a definição de Rede de Petri Temporizada de Alto-Nível.

O desenvolvimento prático do simulador para RPTAN é descrito no capítulo 8, onde são dados a operação do Simulador RPTAN e a explicação de suas estruturas de dados e algoritmos implementados para ele.

Exemplos de modelagem em RPTAN são dados no capítulo 9.

Ao final desta dissertação, encontram-se as conclusões deste trabalho, um apêndice relativo à teoria dos grafos e as referências bibliográficas.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

CAPÍTULO 1: FUNDAMENTOS BÁSICOS DE REDES DE PETRI

1.1. CRONOLOGIA

Redes de Petri (RP) foram criadas a partir da tese de doutorado do pesquisador alemão Carl Adam Petri, intitulada "*Kommunication mit Automaten*" ("Comunicação com Autômatos"). Desde o princípio, RP objetivaram a modelagem de sistemas com componentes concorrentes.

Segundo Heuser [9], as primeiras aplicações de RP aconteceram em 1968, no projeto norte-americano *Information System Theory,* da *A.D.R. - Applied Data Research, Inc.*. Muito da teoria inicial, da notação e da representação de RP foi desenvolvido neste projeto e foi publicado no seu relatório final. Este trabalho ressaltou como RP poderiam ser aplicadas na análise e modelagem de sistemas com componentes concorrentes.

A década de 70 marcou o desenvolvimento da Teoria de RP e a expansão do campo de aplicação da mesma. No início daquela década, o trabalho de Petri chamou a atenção do *Projeto MAC,* do *M.I.T. - Massachusetts Institute of Technology*. O *Grupo de Estruturas Computacionais*, sob a direção de Jack B. Dennis, foi a origem de consideráveis pesquisas e publicações sobre RP, envolvendo relatórios e teses de doutorado. Duas importantes conferências sobre RP foram organizadas pelo grupo: a *Conferência Sobre Sistemas Concorrentes e Computação Paralela*, em Woods Hole (Projeto MAC, 1970), e a *Conferência Sobre Redes de Petri e Métodos Relacionados*, no M.I.T. (1975). Os resultados destes e de outros esforços no desenvolvimento da Teoria de RP estão registrados em inúmeros artigos e em dois livros: no de Reisig [28], que representa o que pode ser considerado a linha européia deste desenvolvimento, e no de Peterson [14], representando a linha norte-americana. Em relação às aplicações, RP atingiu áreas como a modelagem de componentes de *hardware*, controle de processos, linguagens de programação, sistemas distribuídos e protocolos de comunicação.

Ainda na década de 70, surgiram três tipos de RP capazes de modelar diferentes características temporais, quais sejam, as RP Temporizadas de Ramchandani, as de Merlin e as de Sifakis.

As aplicações de RP aumentaram consideravelmente na década de 80, com o surgimento das chamadas Redes de Petri de alto-nível (por exemplo, as Coloridas e as Numéricas). Tais redes acrescentaram uma grande força descritiva ao processo de modelagem, através do uso de *tokens* com identidade e, conseqüentemente, do uso de conjuntos de *tokens*. Desta forma, RP atingiram outras áreas, como automação de escritórios, bancos de dados, inteligência artificial e sistemas de informação.

As RP têm como principal representante, na década de 90, a segunda versão das redes Coloridas, desenvolvidas por Kurt Jensen, da *Århus Universitet*. Além de trabalharem com *tokens* diferenciáveis, tais redes apresentam tratamento de aspectos temporais.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

1.2. Composição Básica de uma Rede de Petri

A seguir, é fornecida a composição básica de uma RP sem considerar, por enquanto, qualquer interpretação ou regra de funcionamento dada à mesma.

Uma RP é representada por um grafo (ver-se-á adiante que pode ser representada matematicamente por meio de matrizes), sendo composta de elementos de dois tipos: **lugares**, indicados por nós elípticos ou circulares, e **transições** entre lugares, indicadas por nós retangulares ou em forma de barra. Ramos orientados ligam lugares às transições ou vice-versa, nunca entre elementos do mesmo conjunto. Inexistem nós lugares ou nós transições isolados. Portanto, de acordo com o exposto, uma RP é um grafo orientado bipartido conexo.

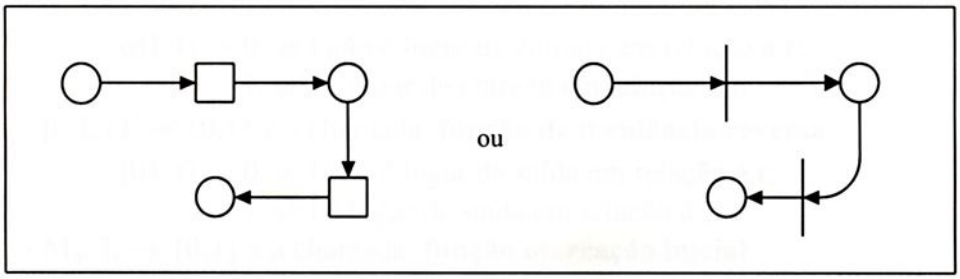


Fig. 1.1. Redes de Petri Equivalentes: 2 Lugares e 2 Transições.

Em relação a uma transição, os ramos são classificados em **ramos de entrada**, indicados pela seta entrando no retângulo, e **ramos de saída**, indicados pela seta saindo do retângulo. Analogamente, classificam-se os lugares como sendo de entrada ou de saída. Um lugar não pode estar ligado a uma transição mais que uma vez através de um mesmo tipo de ramo.

De acordo com o tipo da rede, lugares e transições recebem conotações diferentes. Na maioria das interpretações de redes, os lugares representam elementos do tipo estado e as transições representam elementos do tipo ação.

Esta é a composição básica de uma RP. Na seqüência desta dissertação, serão fornecidos maiores detalhes, de acordo com o tipo de RP estudada.

1.3. Redes de Petri Elementares

Conforme descrito anteriormente, os diversos tipos de RP apresentam diferenças em sua composição e regras de funcionamento. Iniciaremos nossa discussão sobre RP apresentando as características de sua forma clássica, qual seja, a RP Elementar (RPE).

Cada lugar de uma RPE define um estado local ou **condição**, que pode ou não estar vigorando. Graficamente, um estado local ou condição está vigorando quando o lugar que a define possui uma **marca** (ou ***token***, ou ainda, **senha**, indicada por ●), e não está vigorando caso contrário. Assim, dizemos que um lugar está ou não **marcado** (está ou não com um *token*). Denomina-se **marcação da rede** o conjunto de todos os lugares marcados. Enquanto uma marca modela um **estado local** do sistema, uma marcação modela um **estado global**.

unesp

Cada transição de uma RPE define um **evento**, que pode ocorrer frente a determinadas marcações e cuja ocorrência tem por efeito o aparecimento e coincidente desaparecimento de marcas. A **ocorrência de um evento** (também conhecida por **disparo de uma transição**) depende somente dos lugares de entrada da respectiva transição e afeta somente os lugares de saída da transição.

Peterson [14] apresenta a seguinte representação matemática para RPE:

Definição 1.1: **Representação Matemática de RPE.**

Uma RPE pode ser representada pela quíntupla $(\mathbf{L},\mathbf{T},\alpha,\beta,\mathbf{M_0})$, onde:

- **L** é o **conjunto de lugares** da rede;
- **T** é o **conjunto de transições** da rede;
- α: $\mathbf{L}x\mathbf{T} \rightarrow \{0,1\}$ é a chamada **função de incidência direta**

  $\alpha(\mathbf{l}, \mathbf{t})$ = 0, se **l** não é lugar de entrada em relação à **t**;
  = 1, se **l** é lugar de entrada em relação à **t**;
- β: $\mathbf{L}x\mathbf{T} \rightarrow \{0,1\}$ é a chamada **função de incidência reversa**

  $\beta(\mathbf{l}, \mathbf{t})$ = 0, se **l** não é lugar de saída em relação à **t**;
  = 1, se **l** é lugar de saída em relação à **t**;
- $\mathbf{M_0}$: $\mathbf{L} \rightarrow \{0,1\}$ é a chamada **função marcação inicial**

  $\mathbf{M_0}(\mathbf{l})$ = 0, se o lugar **l** não está inicialmente marcado;
  = 1, se o lugar **l** está inicialmente marcado.

Em relação a uma determinada marcação, a possibilidade de ocorrência de um evento segue a regra definida a seguir.

Definição 1.2: **Regra de Habilitação.**

Uma transição está habilitada em relação a uma dada marcação, quando:
a) Seus lugares de entrada, caso a transição os tenha, estão marcados; e
b) Seus lugares de saída, caso a transição os tenha, não estão marcados.

A figura 1.2 apresenta três marcações distintas para uma mesma RP. Na marcação 1.2.a a transição está habilitada, uma vez que todos os seus lugares de entrada estão marcados e seus lugares de saída não estão marcados. A transição não está habilitada na marcação 1.2.b pela falta de um *token* num dos lugares de entrada. A transição também não está habilitada na marcação 1.2.c, devido à presença de um *token* num dos lugares de saída.

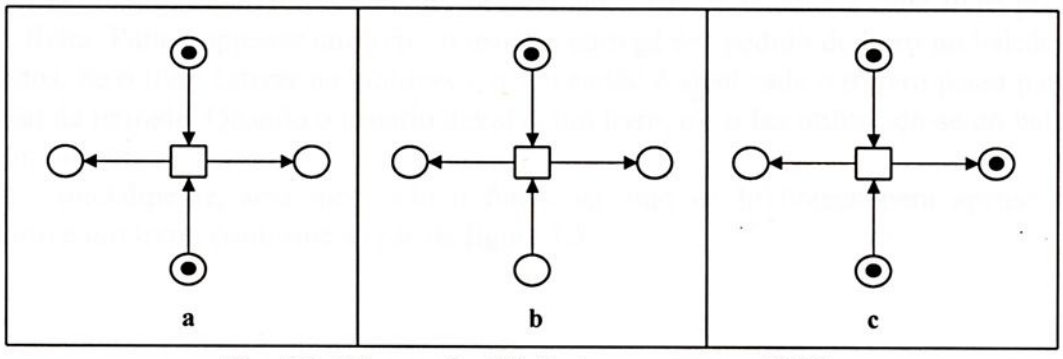


Fig. 1.2. Marcações Distintas para uma RPE.

***Definição 1.3:* Efeito do Disparo de uma Transição.**

O efeito do disparo de uma transição é a passagem de uma marcação **precursora** para uma marcação **sucessora** de tal forma que:

a) As marcas dos lugares de entrada da transição disparada desapareçam na marcação precursora; e
b) É adicionada, na marcação sucessora, uma marca a cada lugar de saída da transição disparada.

A partir desse ponto, é possível compreender o comportamento de uma rede a partir de sua **marcação inicial**, a qual é dada pela função marcação inicial $\mathbf{M_0}$, representada graficamente pela presença e ausência de *tokens* na rede.

***Definição 1.4:* Marcação Alcançável (acessível).**

Dada uma **marcação inicial** da rede, define-se **marcação alcançável** como sendo qualquer marcação obtida pelo efeito da ocorrência de um ou mais disparos sobre a marcação inicial ou sobre qualquer marcação alcançável.

Para representar graficamente as marcações alcançáveis e os disparos das transições responsáveis pelo surgimento das mesmas, utiliza-se a chamada **máquina de senhas**, definida a seguir.

***Definição 1.5:* Máquina de Senhas.**

Máquina de senhas é um grafo que mostra todas as marcações alcançáveis numa RP, a partir de uma dada marcação inicial. Nas RPE, cada **nó** deste grafo representa uma marcação acessível e cada **arco direcionado** representa o disparo da transição que faz a rede passar de uma marcação (a marcação ligada na **origem** do arco) à outra (a marcação ligada no **destino** do arco).

1.4. Exemplo de Modelagem de uma Biblioteca em RPE

Neste, e no próximo capítulo, será utilizado um exemplo-padrão com a finalidade de destacar as diferenças existentes entre as RP Elementares e as RP Lugar/Transição. Tal exemplo consiste na modelagem da organização de uma biblioteca quanto ao empréstimo e à devolução de livros. O usuário pode acessar a biblioteca através de 3 balcões de atendimento, quais sejam: o balcão de **pedidos**, o de **retirada** e o de **devolução**. Os livros encontram-se guardados nas prateleiras e cada livro possui uma **ficha**. Para emprestar um livro, o usuário entrega seu pedido de livro no balcão de pedidos. Se o livro estiver na biblioteca, o seu cartão é atualizado e o livro passa para o balcão de retirada. Quando o usuário devolve um livro, ele o faz utilizando-se do balcão de devolução.

Inicialmente, será modelado o funcionamento da biblioteca para apenas um usuário e um livro, conforme a rede da figura 1.3.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

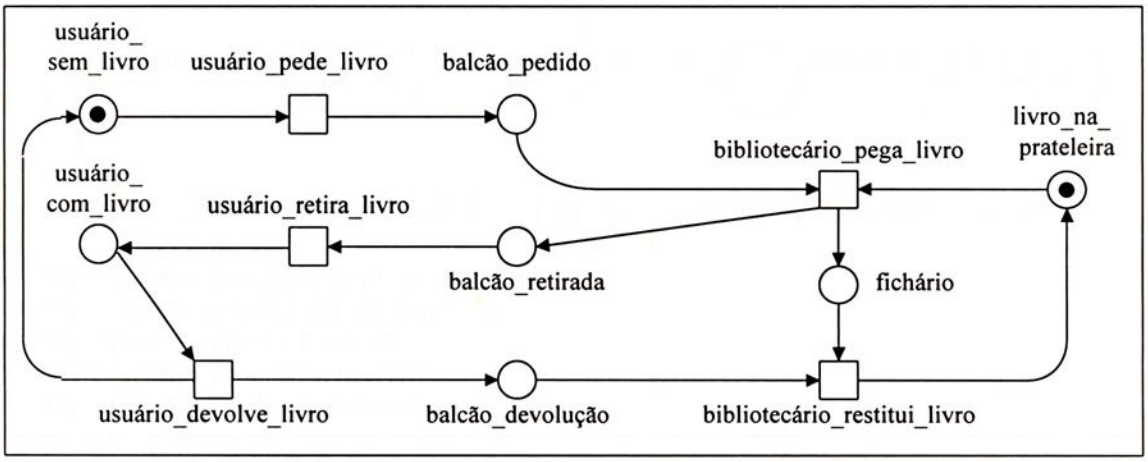


Fig. 1.3. Rede de Petri Elementar Modelando uma Biblioteca com 1 Livro.

Conforme dito anteriormente, a marcação inicial da rede é a denotada no gráfico da mesma. Assim, a RPE da figura 1.3 tem como marcação inicial os lugares **usuário_sem_livro** e **livro_na_prateleira** marcados, representando que o usuário está sem livro e que o livro encontra-se na prateleira.

Inicialmente, a única ação possível é o usuário pedir o livro, o que é feito disparando-se a transição **usuário_pede_livro** (observe que todas as demais transições encontram-se desabitadas). Após o disparo dessa transição, a rede encontra-se na marcação apresentada na figura 1.4.

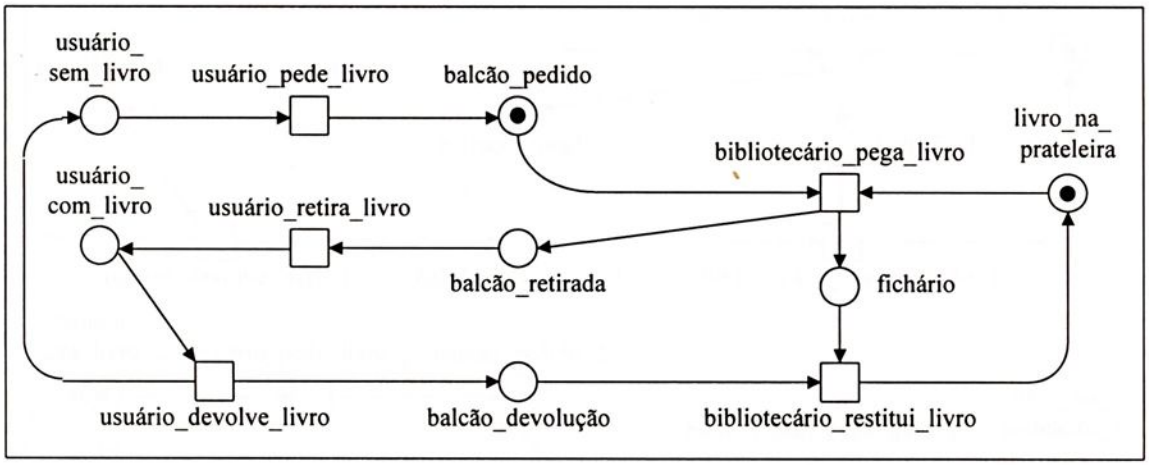


Fig. 1.4. Marcação Decorrente da Inicial Após o Disparo de *usuário_pede_livro*.

Na marcação presente na rede da figura 1.4, a única transição habilitada é **bibliotecário_pega_livro**, a qual representa, no sistema modelado, a retirada do livro da prateleira, a atualização de sua ficha e a entrega do mesmo no balcão de retirada.

Pode-se visualizar melhor o comportamento da rede através de sua máquina de senhas, dada na figura 1.5. Nela, vê-se as possíveis marcações da rede e a relação entre elas, ou seja, qual transição cujo disparo leva o modelo de uma marcação à outra.

unesp

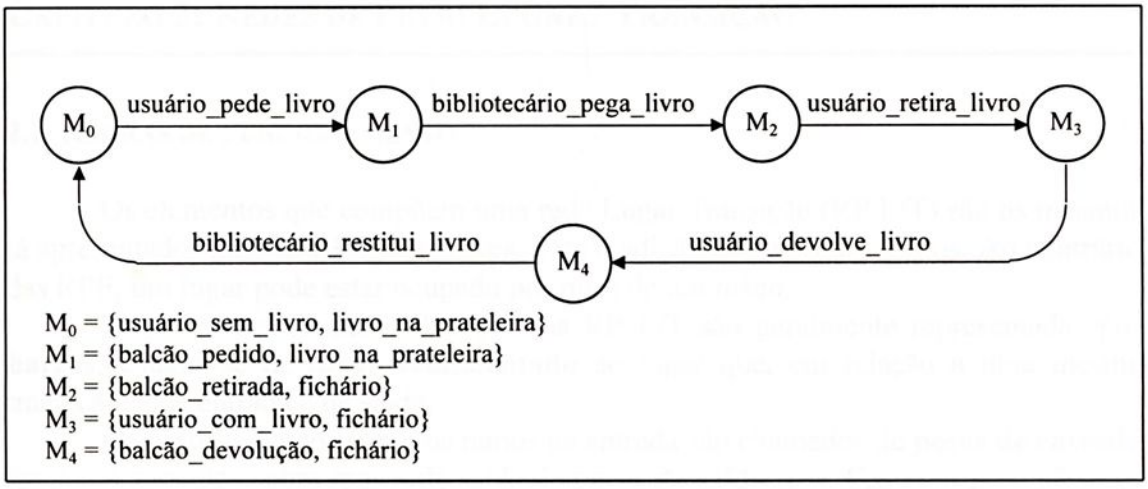


Fig. 1.5. Máquina de Senhas da RPE da Figura 1.3.

Certamente, não se deseja modelar uma biblioteca que possua apenas um livro. A RPE da figura 1.6 modela uma biblioteca com 2 livros. Conforme pode-se observar nessa figura, houve a necessidade de duplicar a rede da figura 1.3 para modelar uma biblioteca com 2 livros. Portanto, RPE mostram-se inadequadas para o tratamento de sistemas onde o aspecto quantitativo seja importante. O próximo capítulo apresenta uma melhoria do modelo representativo da biblioteca, por meio das RP Lugar/Transição.

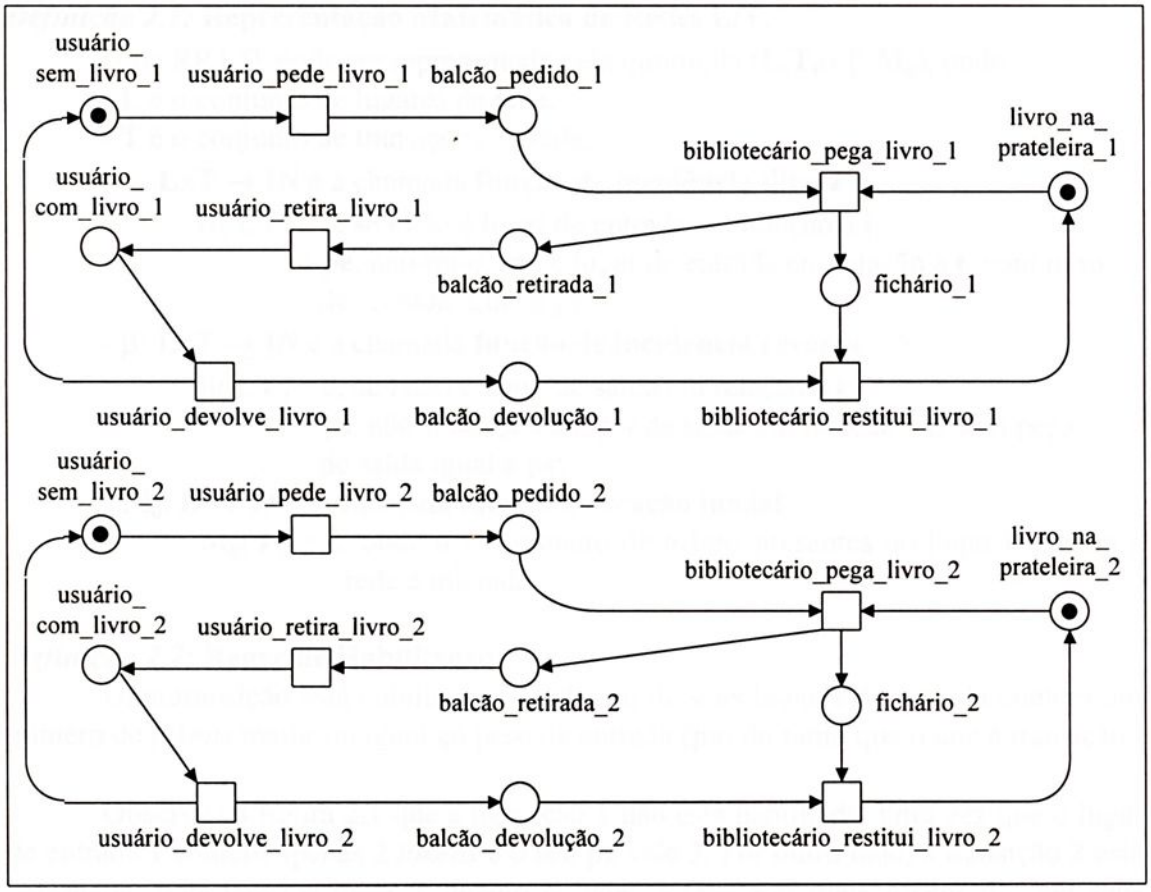


Fig. 1.6. Rede de Petri Elementar Modelando uma Biblioteca com 2 Livros.

CAPÍTULO 2: REDES DE PETRI LUGAR / TRANSIÇÃO

2.1. REGRAS DE FUNCIONAMENTO

Os elementos que compõem uma rede Lugar/Transição (RP L/T) são os mesmos já apresentados para as RP Elementares, com a adição de pesos aos ramos. Ao contrário das RPE, um lugar pode estar ocupado por mais de um *token*.

Na literatura, as transições de uma RP L/T são geralmente representadas por **barras**. Chama-se de **lugar realimentado** ao lugar que, em relação a uma mesma transição, é de entrada e de saída.

Os pesos anotados sobre os ramos de entrada são chamados de **pesos de entrada** (**pe**) e os anotados sobre ramos de saída de **pesos de saída** (**ps**). Caso um peso não seja especificado, subentende-se que o mesmo é 1. O peso **pe** representa o número de *tokens* que o lugar de entrada perde, caso haja o disparo da transição associada. Por outro lado, o peso **ps** representa o número de *tokens* que o lugar de saída ganha, caso haja o disparo da transição correspondente.

Segue-se a formalização matemática das RP L/T [25], a qual ajudará no tratamento computacional dos diversos tipos de RP. Nestas definições, **IN** representa o conjunto dos números naturais.

Definição 2.1: **Representação Matemática de Redes L/T.**

Uma RP L/T pode ser representada pela quíntupla $(\mathbf{L},\mathbf{T},\alpha,\beta,\mathbf{M_0})$, onde:

- **L** é o conjunto de lugares da rede;
- **T** é o conjunto de transições da rede;
- α: $\mathbf{L}x\mathbf{T} \rightarrow \mathbf{IN}$ é a chamada **função de incidência direta**

  $\alpha(\mathbf{l}, \mathbf{t}) = 0$, se **l** não é lugar de entrada em relação à **t**;
  $= \mathbf{pe}$, não-nulo, se **l** é lugar de entrada em relação à **t**, com peso de entrada igual a **pe**;
- β: $\mathbf{L}x\mathbf{T} \rightarrow \mathbf{IN}$ é a chamada **função de incidência reversa**

  $\beta(\mathbf{l}, \mathbf{t}) = 0$, se **l** não é lugar de saída em relação à **t**;
  $= \mathbf{ps}$, não-nulo, se **l** é lugar de saída em relação à **t**, com peso de saída igual a **ps**;
- $\mathbf{M_0}$: $\mathbf{L} \rightarrow \mathbf{IN}$ é a chamada **função marcação inicial**

  $\mathbf{M_0}(\mathbf{l}) = \mathbf{n}$, onde **n** é o número de *tokens* presentes no lugar **l** quando a rede é iniciada.

Definição 2.2: **Regra de Habilitação.**

Uma transição está habilitada se cada um de seus lugares de entrada contém um número de *tokens* maior ou igual ao peso de entrada (**pe**) do ramo que o une à transição.

Observe na figura 2.1 que a transição **1** não está habilitada, uma vez que o lugar de entrada **1** contém apenas 2 *tokens* e o seu **pe** vale 3. Por outro lado, a transição **2** está habilitada, pois tanto o lugar **2** quanto o lugar **3** possuem mais *tokens* do que seus respectivos **pe**´s.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

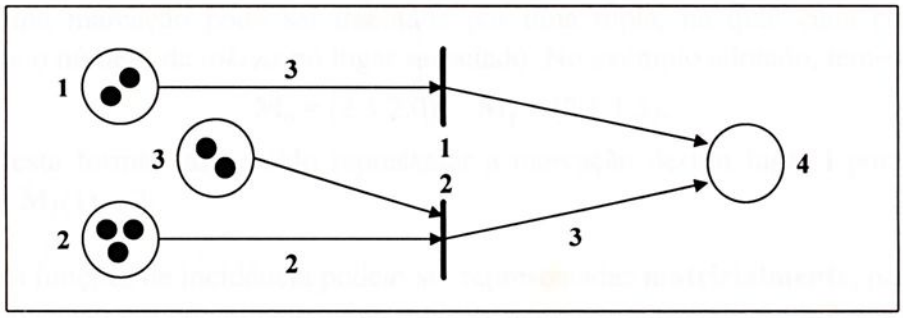


Fig. 2.1. Transição 2 Habilitada e Transição 1 Desabilitada.

Definição 2.3: **Efeito do Disparo de uma Transição.**

O disparo de uma transição **t**, habilitada, consiste em remover **pe** *tokens* de cada um dos lugares de entrada e adicionar **ps** *tokens* a cada um dos lugares de saída. Diz-se que a rede passou de uma marcação $\mathbf{M_i}$ para uma marcação $\mathbf{M_j}$ através do disparo da transição **t**.

Definição 2.4: **Conjunto de Marcações Decorrentes de $\mathbf{M_0}$.**

É o conjunto de todas as marcações pelas quais o sistema pode passar, partindo de uma dada marcação inicial $\mathbf{M_0}$.

Notação: $\{\mathbf{M_0}\}$.

O disparo da transição **2** da rede da figura 2.1 faz a mesma passar para a marcação denotada na figura 2.2. Temos a seguinte notação para representar esta mudança entre as marcações:

$$\mathbf{M_0} \xrightarrow{2} \mathbf{M_1}$$

A notação anterior pode ser estendida para uma seqüência de disparos.

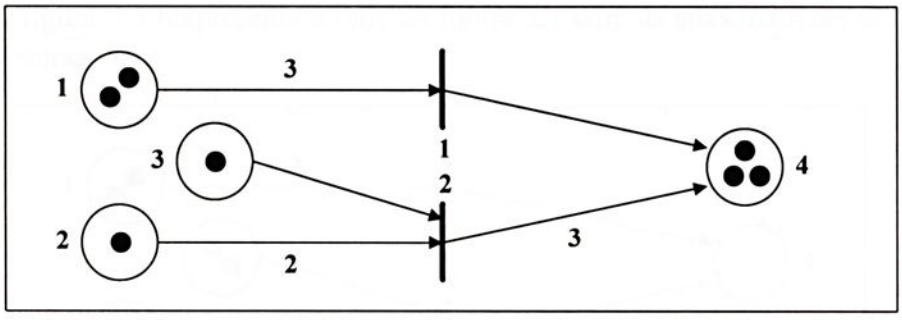


Fig. 2.2. Marcação Alcançada com o Disparo da Transição 2.

Definição 2.5: **Seqüência de Disparos (ou σ).**

É um conjunto de transições disparadas em seqüência, responsável pela passagem da rede de uma marcação $\mathbf{M_i}$ a uma marcação $\mathbf{M_j}$, não necessariamente distintas.

Notação: $\mathbf{M_i} \xrightarrow{\sigma} \mathbf{M_j}$, onde σ é a seqüência de disparos.

unesp

Uma marcação pode ser denotada por uma tupla, na qual cada componente representa o número de *tokens* no lugar associado. No exemplo adotado, tem-se:

$$\mathbf{M_0} = (2,3,2,0) \quad \mathbf{M_1} = (2,1,1,3).$$

Desta forma, faz sentido representar a marcação de um lugar **l** por **M(l)**. Por exemplo, $\mathbf{M_1}(1) = 2$.

As funções de incidência podem ser representadas **matricialmente**, para facilitar sua manipulação em computador. São duas as matrizes: a **matriz de incidência direta** (relativa a α) e a **matriz de incidência reversa** (relativa a β).

Definição 2.6: **Matriz de Incidência Direta (ou Matriz A).**

É uma matriz formada por #**T** linhas e #**L** colunas, representante da **estrutura da rede** quanto às informações dos lugares de entrada que incidem sobre cada transição e seu respectivo peso de entrada.

O elemento da linha **i** e coluna **j** (**i** = 1,...,#**T**; **j** = 1,...,#**L**), denotado por $\mathbf{A_{ij}}$, contém as seguintes informações:

a) se $\mathbf{A_{ij}} = 0$, o lugar $\mathbf{l_j}$ não é lugar de entrada da transição $\mathbf{t_i}$;

b) se $\mathbf{A_{ij}} \neq 0$, o lugar $\mathbf{l_j}$ é um lugar de entrada da transição $\mathbf{t_i}$, com $\mathbf{A_{ij}}$= **pe**.

Definição 2.7: **Matriz de Incidência Reversa (ou Matriz B).**

É uma matriz formada por #**T** linhas e #**L** colunas, representante da estrutura da rede quanto às informações dos lugares de saída que são atingidos por cada transição e seu respectivo peso de saída.

O elemento da linha **i** e coluna **j** (**i** = 1,...,#**T**; **j** = 1,...,#**L**), denotado por $\mathbf{B_{ij}}$, contém as seguintes informações:

a) se $\mathbf{B_{ij}} = 0$, o lugar $\mathbf{l_j}$ não é lugar de saída da transição $\mathbf{t_i}$;

b) se $\mathbf{B_{ij}} \neq 0$, o lugar $\mathbf{l_j}$ é um lugar de saída da transição $\mathbf{t_i}$, com $\mathbf{B_{ij}}$= **ps**.

A figura 2.3 reapresenta a rede da figura 2.1 com as suas matrizes de incidência direta e reversa.

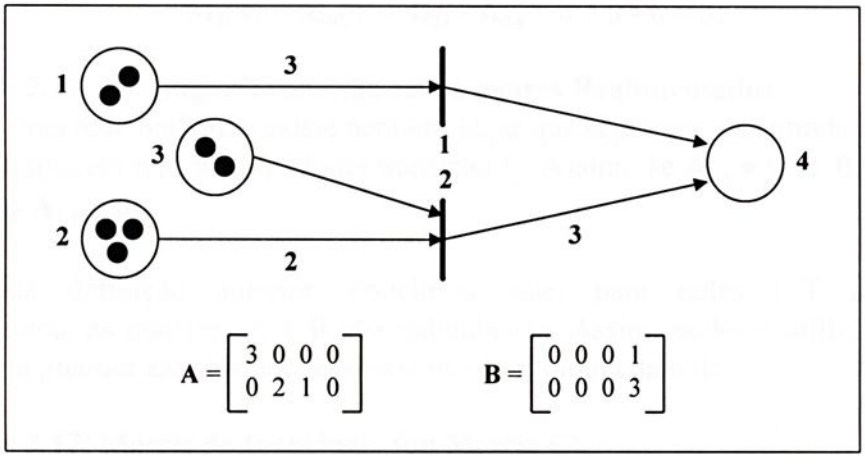


Fig. 2.3. Matrizes de Incidência Direta (A) e Reversa (B).

Definidas as matrizes associadas a uma rede L/T, pode-se fazer as definições matemáticas a seguir.

Definição 2.8: **Conjunto de Lugares de Entrada e de Saída de uma Transição.**

O conjunto de lugares de entrada da transição $\mathbf{t_i}$ é definido por:

$$\mathbf{E(t_i) = \{ l_j \in L \mid A_{ij} \neq 0 \}\ (i = 1,...,\#T;\ j = 1,...,\#L).}$$

O conjunto de lugares de saída da transição $\mathbf{t_i}$ é definido por:

$$\mathbf{S(t_i) = \{ l_j \in L \mid B_{ij} \neq 0 \}\ (i = 1,...,\#T;\ j = 1,...,\#L).}$$

Por exemplo, na figura 2.1, tem-se:

$$\mathbf{E}(1) = \{1\},\ \mathbf{S}(1) = \{4\},\ \mathbf{E}(2) = \{2,3\},\ \mathbf{S}(2) = \{4\}.$$

Definição 2.9: **Regra de Habilitação (representação matemática).**

Uma transição $\mathbf{t_i}$ é habilitada ou disparável em uma dada marcação **M**, se e somente se:

$$\forall\ \mathbf{l_j} \in \mathbf{E(t_i)},\ \mathbf{M(l_j)} \geq \mathbf{A_{ij}}\ \ (\mathbf{i} = 1,...,\#\mathbf{T};\ \mathbf{j} = 1,...,\#\mathbf{L}).$$

Na figura 2.3, apenas a transição **2** pode disparar, pois:

$$\mathbf{E}(1) = \{1\} \text{ e } \mathbf{M_0}(1) = 2 < \mathbf{A_{11}} = 3; \text{ e}$$
$$\mathbf{E}(2) = \{2,3\} \text{ e } \mathbf{M_0}(2) = 3 > \mathbf{A_{22}} = 2 \text{ e } \mathbf{M_0}(3) = 2 > \mathbf{A_{23}} = 1.$$

Definição 2.10: **Efeito do Disparo de uma Transição (representação matemática).**

O disparo de uma transição habilitada $\mathbf{t_i}$ é a passagem da rede de uma marcação **M** à outra **M'**, tal que:

$$\forall\ \mathbf{l_j} \in \mathbf{L},\ \mathbf{M'(l_j) = M(l_j) + B_{ij} - A_{ij}}\ \ (\mathbf{i} = 1,...,\#\mathbf{T};\ \mathbf{j} = 1,...,\#\mathbf{L}).$$

Conforme visto na figura 2.3, para $\mathbf{M_0} = (2,3,2,0)$, o disparo da transição **2** resultará em $\mathbf{M_1}$, tal que:

$$\mathbf{M_1}(1) = \mathbf{M_0}(1) + \mathbf{B_{21}} - \mathbf{A_{21}} = 2 + 0 - 0 = 2;$$
$$\mathbf{M_1}(2) = \mathbf{M_0}(2) + \mathbf{B_{22}} - \mathbf{A_{22}} = 3 + 0 - 2 = 1;$$
$$\mathbf{M_1}(3) = \mathbf{M_0}(3) + \mathbf{B_{23}} - \mathbf{A_{23}} = 2 + 0 - 1 = 1;$$
$$\mathbf{M_1}(4) = \mathbf{M_0}(4) + \mathbf{B_{24}} - \mathbf{A_{24}} = 0 + 3 - 0 = 3;$$

Definição 2.11: **RP Lugar/Transição sem Lugares Realimentados.**

É uma rede onde não existe nenhum lugar que seja lugar de entrada e de saída ao mesmo tempo, em relação a qualquer transição $\mathbf{t_i}$. Assim, se $\mathbf{A_{ij} \neq 0 \Rightarrow B_{ij} = 0}$, ou, se $\mathbf{B_{ij} \neq 0 \Rightarrow A_{ij} = 0}$.

Pela definição anterior, conclui-se que, para redes L/T sem lugares realimentados, as matrizes **A** e **B** são redundantes. Assim, pode-se utilizar uma única matriz para guardar as informações referentes à topologia da rede.

Definição 2.12: **Matriz de Incidência (ou Matriz C).**

É a matriz definida pela subtração da matriz **B** pela **A**, isto é, **C = B - A**. A matriz **C** contém as seguintes informações:

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

a) se $C_{ij} = 0$, o lugar l_j não é nem lugar de entrada e nem lugar de saída da transição t_i;
b) se $C_{ij} > 0$, o lugar l_j é um lugar de entrada da transição t_i, com $C_{ij} = pe$;
c) se $C_{ij} < 0$, o lugar l_j é um lugar de saída da transição t_i, com $C_{ij} = ps$.

Deixa-se observado que a definição de máquina de senhas para as redes L/T é a mesma que a apresentada para as RP Elementares (definição 1.5). Observa-se, também, que as propriedades a serem descritas a seguir são geralmente semelhantes para as demais extensões de RP.

2.2. Propriedades Gerais de uma RP Lugar/Transição

2.2.1. Limitabilidade

Uma rede L/T é definida como **limitada** a um valor natural **n** se, para qualquer marcação **M** decorrente da marcação inicial M_0, o número de *tokens* em qualquer lugar não ultrapassa **n**. Matematicamente, tem-se:

$$\forall\, M \in \{M_0\},\ \forall\, l_j \in L,\ M(l_j) \leq n\ \ (j = 1,...,\#L).$$

Recebe o nome de **segura**, a RP L/T limitada a $n = 1$.

Numa rede limitada, o número de marcações é **finito**. O mesmo não ocorre com uma rede **ilimitada**, significando a impossibilidade de implementação do sistema modelado.

A rede da figura 2.4 é ilimitada, pois se uma seqüência de disparos $\sigma = (1,2,3)$ for repetida infinitas vezes, implicará em $M_i(1)$ sempre crescente.

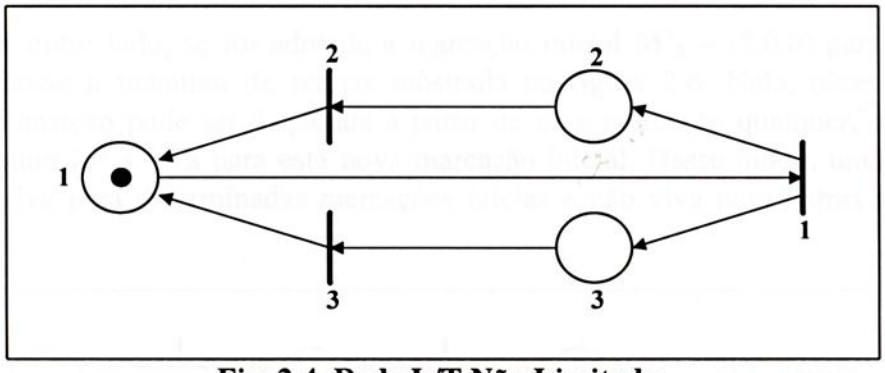


Fig. 2.4. Rede L/T Não Limitada.

2.2.2. Vivacidade

Uma rede L/T é **viva** para uma marcação inicial M_0, se para qualquer marcação decorrente **M** e para cada transição **t** considerada, existir uma seqüência de disparos σ, de forma que essa transição seja habilitada.

Matematicamente, tem-se:

$$\forall\, M \in \{M_0\},\ \forall\, t \in T,\ \exists\, \sigma \mid (\, M \xrightarrow{\sigma'} M' \ \wedge\ M' \xrightarrow{t} M''\,)$$

onde $\sigma = (\sigma', t)$ e $M', M'' \in \{M_0\}$.

Um sistema modelado por uma rede L/T viva está livre de *deadlocks*, isto é, não possui situações de impasse.

Considere a rede da figura 2.5.a. Para a marcação inicial $\mathbf{M_0} = (1,0,0)$, a máquina de senhas correspondente é mostrada na figura 2.5.b.

A transição **3** nunca irá disparar para a marcação $\mathbf{M_0} = (1,0,0)$, pois tem peso de entrada igual a **2**. Observe que a transição **4** também não dispara. Conclusão: **inexiste** uma seqüência de disparos que faça a rede alcançar uma marcação na qual **3** possa disparar. Assim, a rede da figura 2.5.a não é viva para a marcação inicial adotada.

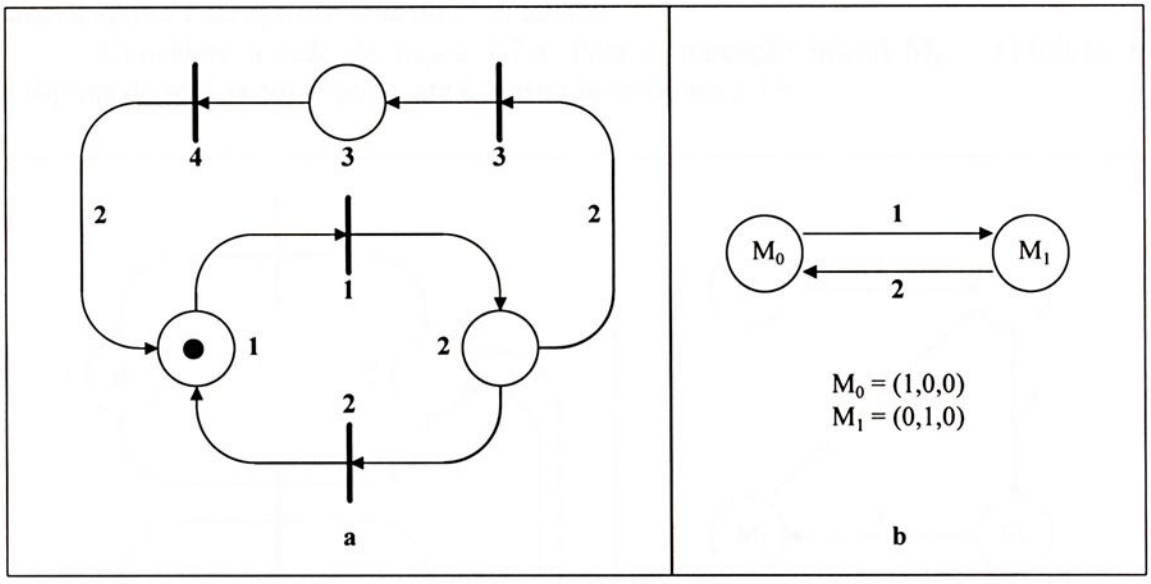


Fig. 2.5. Rede L/T Não Viva e sua Máquina de Senhas.

Por outro lado, se for adotada a marcação inicial $\mathbf{M'_0} = (2,0,0)$ para a mesma rede, obtém-se a máquina de senhas mostrada na figura 2.6. Nela, observa-se que qualquer transição pode ser disparada a partir de uma marcação qualquer, tornando a rede da figura 2.5.a viva para esta nova marcação inicial. Desse modo, uma rede L/T pode ser viva para determinadas marcações inicias e não viva para outras marcações inicias.

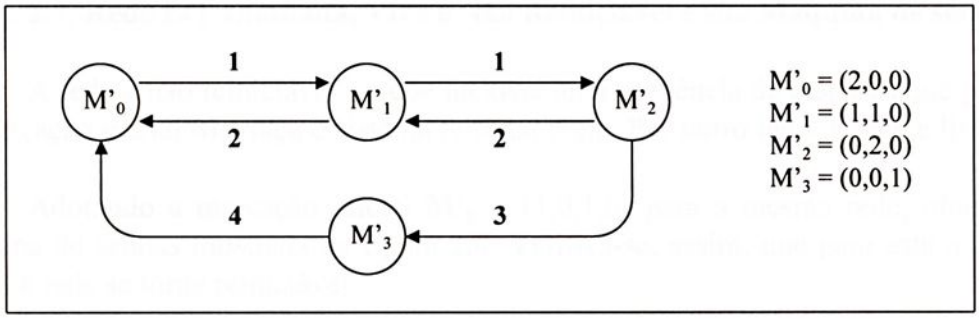


Fig. 2.6. Máquina de Senhas da Rede da Figura 2.5;
Rede Viva para Nova Marcação Inicial.

2.2.3. REINICIABILIDADE

Uma RP L/T é **reiniciável** para uma dada marcação inicial $\mathbf{M_0}$, se para qualquer marcação decorrente $\mathbf{M} \in \{\mathbf{M_0}\}$ existir uma seqüência de disparos σ tal que faça a rede voltar à marcação inicial $\mathbf{M_0}$.

Matematicamente, tem-se:

$$\forall\, \mathbf{M} \in \{\mathbf{M_0}\},\ \exists\, \sigma \mid (\mathbf{M} \xrightarrow{\sigma} \mathbf{M_0}).$$

Um sistema físico modelado por uma rede reiniciável **retorna** ao seu estado inicial após a execução de uma ou mais tarefas.

Considere a rede da figura 2.7.a. Para a marcação inicial $\mathbf{M_0}$ = (1,0,0,1), a máquina de senhas correspondente é mostrada na figura 2.7.b.

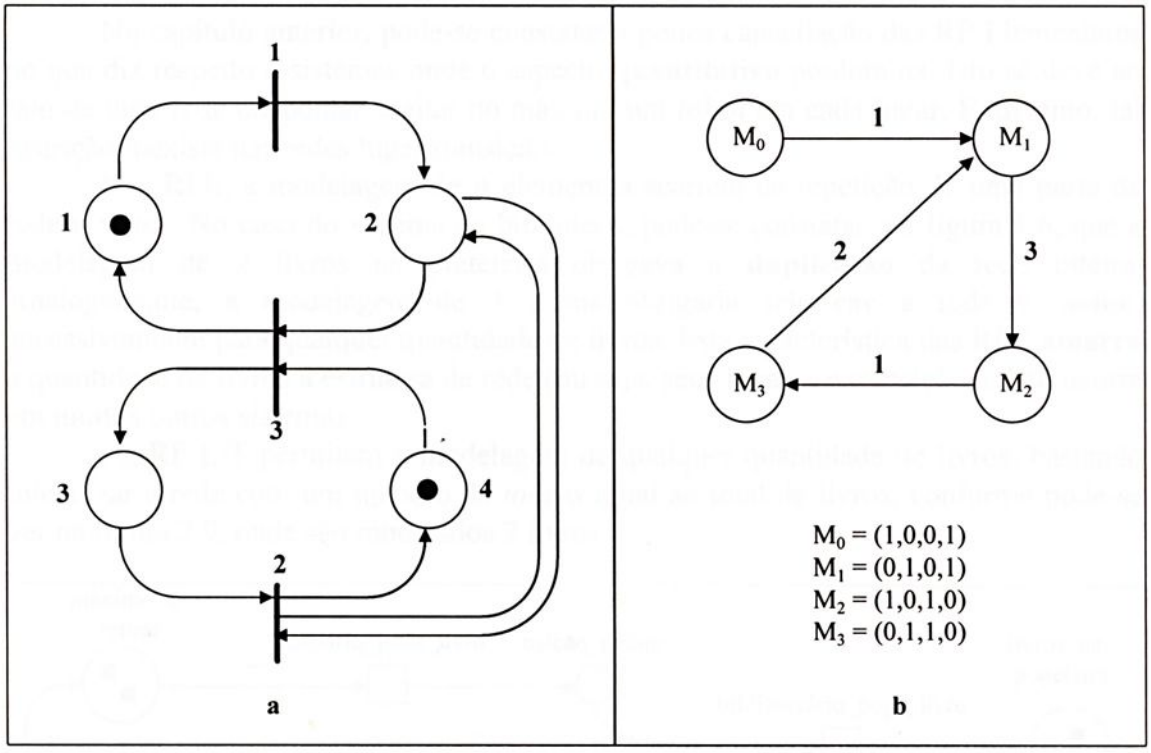


Fig. 2.7. Rede L/T Limitada, Viva e Não Reiniciável e sua Máquina de senhas.

A rede é não reiniciável porque inexiste uma seqüência de disparos que partindo da marcação inicial $\mathbf{M_0}$, faça o sistema retornar a ela. Por outro lado, a rede é limitada e viva.

Adotando a marcação inicial $\mathbf{M'_0}$ = (1,0,1,0) para a mesma rede, obtém-se a máquina de senhas mostrada na figura 2.8. Verifica-se, assim, que para esta marcação inicial a rede se torna reiniciável.

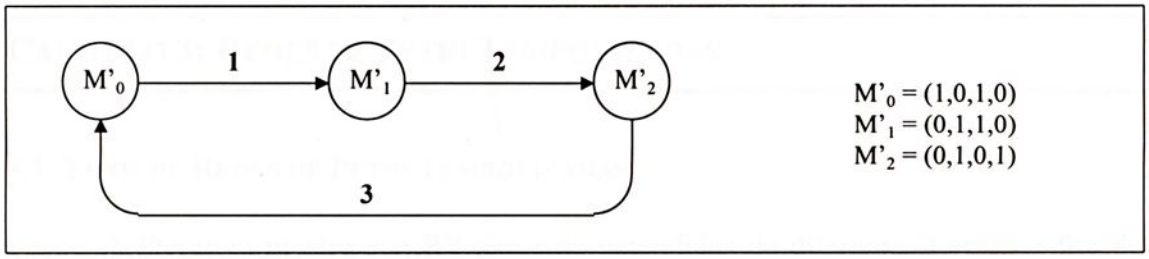


Fig. 2.8. Máquina de Senhas da Rede da Figura 2.7.a com Nova Marcação Inicial.

2.3. EXEMPLO DE MODELAGEM DE UMA BIBLIOTECA EM RP L/T

Nesta seção, será retomado o exemplo do sistema que consiste no atendimento de usuários em uma biblioteca, descrito na seção 1.4.

No capítulo anterior, pode-se constatar a pouca capacitação das RP Elementares no que diz respeito a sistemas onde o aspecto **quantitativo** predomina. Isto se deve ao fato de uma rede elementar aceitar no máximo um *token* em cada lugar. Entretanto, tal restrição inexiste nas redes lugar/transição.

Nas RPE, a modelagem de **n** elementos acarreta na repetição de uma parte da rede **n** vezes. No caso do sistema da biblioteca, pode-se constatar, na figura 1.6, que a modelagem de 2 livros na prateleira obrigava a **duplicação** da rede inteira. Analogamente, a modelagem de 3 livros obrigaria **triplicar** a rede e, assim, sucessivamente para qualquer quantidade de livros. Esta característica das RPE **amarra** a quantidade de livros à estrutura da rede (ou seja, seus lugares e transições). Isto ocorre em muitos outros sistemas.

As RP L/T permitem a modelagem de qualquer quantidade de livros, bastando inicializar a rede com um número de *tokens* igual ao total de livros, conforme pode-se ver na figura 2.9, onde são modelados 2 livros.

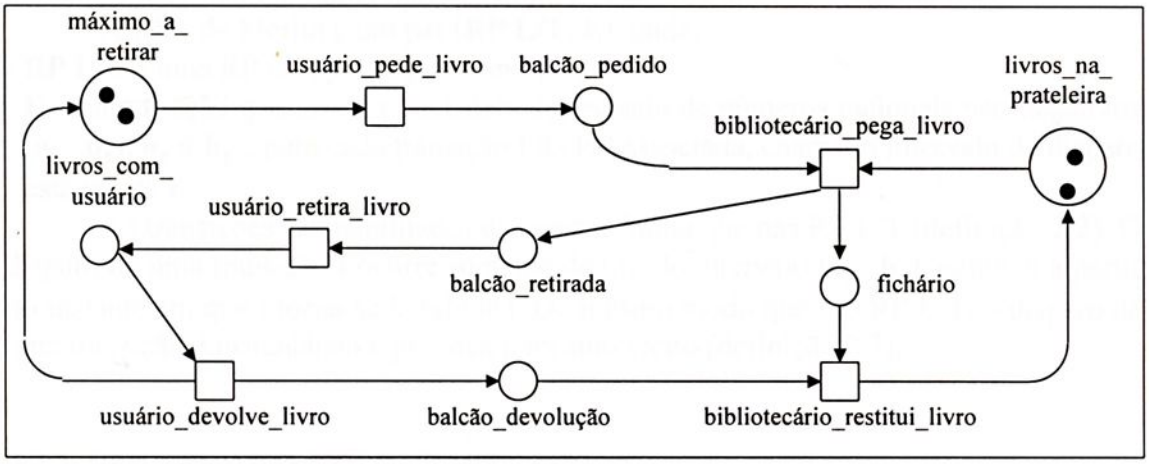


Fig. 2.9. RP L/T Modelando uma Biblioteca com 2 Livros.

Agora, tem-se uma situação onde o usuário pode retirar até dois livros. Apesar de inadequada, a RPE da figura 1.6 diferenciava os 2 livros. Assim, RP L/T são adequadas somente quando não necessita-se distinguir entre a **natureza** dos *tokens* envolvidos, importando-nos somente a quantidade dos mesmos. As redes de alto-nível, cuja principal representante são as Coloridas, surgiram para resolver este problema.

CAPÍTULO 3: REDES DE PETRI TEMPORIZADAS

3.1. TIPOS DE REDES DE PETRI TEMPORIZADAS

A literatura mostra que RP têm sido extendidas de diferentes formas a fim de incorporar o conceito de **tempo**. Tais extensões surgiram para o modelamento de sistemas temporizados com características diferentes. Segundo Ghezzi et al. [8], tais extensões podem ser classificadas em duas grandes famílias, quais sejam:

a) As que associam valores de tempo às transições; e
b) As que associam valores de tempo aos lugares.

A seguir, será dada uma breve descrição destas famílias de Redes de Petri Temporizadas (RPT) [5,8,23]. Na seção 3.4, serão vistos maiores detalhes sobre as RPT de Merlin, pois, segundo Marton et al. [22], os demais tipos de RPT são casos particulares do modelo proposto por Merlin.

3.2. RP TEMPORIZADAS COM VALORES DE TEMPO ASSOCIADOS ÀS TRANSIÇÕES

Há duas formas de associar tempo às transições de uma RP. Uma dessas formas encontra-se representada nas RPT de Merlin, enquanto a outra forma tem como representante principal as RPT de Ramchandani.

3.2.1. DESCRIÇÃO DAS RPT DE MERLIN

Esta extensão (Time Petri Nets - TPN) foi proposta por Merlin em 1974. Neste modelo, é associado um intervalo de tempo a cada uma das transições da rede.

A RPT de Merlin é um par (**RP L/T**, **E**), onde:

- **RP L/T** é uma RP do tipo Lugar/Transição;
- **E** é uma função que associa um intervalo fechado de números racionais não negativos $[a_t, b_t]$, $a_t \leq b_t$, para cada transição **t** da RP associada, chamado intervalo de disparo estático de **t**.

As transições são habilitadas da mesma forma que nas RP L/T (definição 2.2). O disparo de uma transição **t** ocorre somente dentro do intervalo $[a_t, b_t]$, contado a partir do instante em que **t** torna-se habilitada. Do mesmo modo que nas RP L/T, o disparo de uma transição é instantâneo e provoca o mesmo efeito (definição 2.3).

3.2.2. DESCRIÇÃO DAS RPT DE RAMCHANDANI

Esta extensão (Timed Petri Nets - TdPN) foi proposta por Ramchandani em 1973. Neste modelo, é associado um tempo para o disparo de cada transição da rede. Diferentemente das RP originais, o disparo da transição não é mais um evento instantâneo.

unesp

A RPT de Ramchandani é um par (**RP L/T**, **D**), onde:

- **RP L/T** é a Rede de Petri Lugar/Transição ($\mathbf{L,T,\alpha,\beta,M_0}$) sem a consideração dos tempos envolvidos, chamada RP associada;
- **D** é uma função que associa um número relativo à duração do disparo de cada transição da RP associada.

As transições são habilitadas do mesmo modo que nas **RP L/T** (definição 2.2). O disparo de uma transição habilitada **t** causa uma mudança da marcação **M** para a marcação **M''**, através de duas etapas (não mais uma única etapa instantânea):

1) Tão logo a transição **t** seja habilitada, ela deve disparar (a menos que outra transição dispare e a desabilite), promovendo a retirada de *tokens* de seus lugares de entrada, isto é, $\forall\, \mathbf{l} \in \mathbf{L},\ \mathbf{M'}(\mathbf{l}) = \mathbf{M}(\mathbf{l}) - \alpha(\mathbf{l}, \mathbf{t})$, onde:
   M(**l**) é o número de *tokens* no lugar **l** antes da realização dessa primeira etapa;
   M'(**l**) é o número de *tokens* no lugar **l** após a realização dessa primeira etapa;
   α (**l**, **t**) é o peso do arco que vai do lugar **l** para a transição **t**.
2) **D**(**t**) unidades de tempo após o disparo de **t**, *tokens* são inseridos em seus lugares de saída, isto é, $\forall\, \mathbf{l} \in \mathbf{L},\ \mathbf{M''}(\mathbf{l}) = \mathbf{M'}(\mathbf{l}) + \beta(\mathbf{l}, \mathbf{t})$, onde:
   M''(**l**) é o número de *tokens* no lugar **l** após a realização dessa segunda etapa, ou seja, após o disparo completo da transição;
   β(**l**, **t**) é o peso do arco que vai da transição **t** para o lugar **l**.

3.3. RP Temporizadas com Valores de Tempo Associados aos Lugares

A principal extensão desta família foi proposta por Sifakis em 1977 (Timed Place/Transition Nets - TdPTN). Neste modelo, é associado um tempo de permanência de *tokens* em cada lugar da rede.

A RPT de Sifakis é um par (**RP L/T**, **A**), onde:

- **RP L/T** é uma RP do tipo Lugar/Transição;
- **A** é uma função que associa um atraso a cada lugar da RP associada.

Sifakis demonstrou, em 1979, a equivalência de seu modelo com o de Ramchandani.

3.4. RP Temporizadas de Merlin (RPTM)

Sendo o modelo de rede temporizada mais geral, nesta seção será estudada com mais detalhes a definição matemática de RPTM [5,23].

Definição 3.1: **Representação Matemática de RPTM.**

Uma RPTM pode ser representada pela sêxtupla ($\mathbf{L,T,\alpha,\beta,M_0,E}$) , onde:

- **L** é o conjunto de **lugares** da rede;
- **T** é o conjunto de **transições** da rede;
- α: $\mathbf{L \times T \rightarrow IN}$ é a chamada **função de incidência direta**

α(**l**, **t**) = 0, se **l** não é lugar de entrada em relação à **t**
= **pe**, não-nulo, se **l** é lugar de entrada em relação à **t**, com peso

de entrada igual a **pe**;

- β: **LxT → IN** é a chamada **função de incidência reversa**

  β(**l**, **t**) = 0, se **l** não é lugar de saída em relação à **t**;

  = **ps**, não-nulo, se **l** é lugar de saída em relação à **t**, com peso de saída igual a **ps**.

- $\mathbf{M_0}$: **L → IN** é a chamada **função marcação inicial**

  $\mathbf{M_0}$(**l**) = **n**, onde **n** é o número de *tokens* presentes no lugar **l** quando a rede é iniciada;

- **E**: $\mathbf{T \rightarrow (Q^+ \cup \{0\}) \times (Q^+ \cup \{0\} \cup \infty)}$, **Q** conjunto dos números racionais, é a chamada **função de intervalos estáticos.**

  $\mathbf{E(t) = [a_t, b_t]}$, onde $\mathbf{0 \leq a_t \leq b_t}$

Observações da definição 3.1:

a) O limite inferior $\mathbf{a_t}$ é chamado **tempo de disparo inicial (TDI) estático**, ou simplesmente TDI estático;

b) O limite superior $\mathbf{b_t}$ é chamado **tempo de disparo final (TDF) estático**, ou simplesmente TDF estático;

c) O intervalo de disparo [**TDI**, **TDF**] de uma transição **t delimita** o instante de tempo no qual poderá ocorrer o disparo de uma transição, a partir do momento em que ela torna-se habilitada. Pode-se afirmar que RP L/T é um caso particular de RPTM, onde associa-se a cada transição o intervalo [0, ∞].

Definição 3.2: **Marcação de uma RPTM.**

A marcação de uma RPTM é uma tupla de pares ordenados (**l**,**m**), onde **l** é um lugar não vazio dentro da marcação e **m** é o número de *tokens* presentes em **l**.

Definição 3.3: **Função de Intervalos Dinâmicos.**

D: $\mathbf{T \rightarrow (Q^+ \cup \{0\}) \times (Q^+ \cup \{0\} \cup \infty)}$, **Q** conjunto dos números racionais.

$\mathbf{D(t) = [a_t, b_t]}$, onde $\mathbf{0 \leq a_t \leq b_t}$

A função de intervalos dinâmicos **D** associa, a cada transição **t** habilitada por uma marcação **M**, um intervalo de tempo no qual é permitido o disparo de **t**. Este intervalo de tempo é chamado de intervalo de disparo dinâmico, correspondendo ao restante do intervalo no qual **t** ainda pode disparar.

Definição 3.4: **Estado de uma RPTM.**

O **estado** de uma RPTM é dado por $\mathbf{S_i = (M_i, D_i)}$, onde:

a) $\mathbf{M_i}$ é a **marcação** da RP associada;

b) $\mathbf{D_i}$ é a **função de intervalos dinâmicos** para as transições habilitadas em $\mathbf{M_i}$.

A figura 3.1 mostra uma RPTM com estado inicial $\mathbf{S_0 = (M_0, D_0)}$. Nela, tem-se:

a) $\mathbf{M_0}$ = ((1,1), (6,1)), significando que o lugar **1** tem 1 *token* e o lugar **6** tem 1 *token*;

b) $\mathbf{D_0}$ = (1[0,2], 6[2,3]), significando que a transição **1** está habilitada e tem intervalo de disparo dinâmico [0,2] e que a transição **6** também está habilitada, com intervalo de disparo dinâmico [2,3].

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

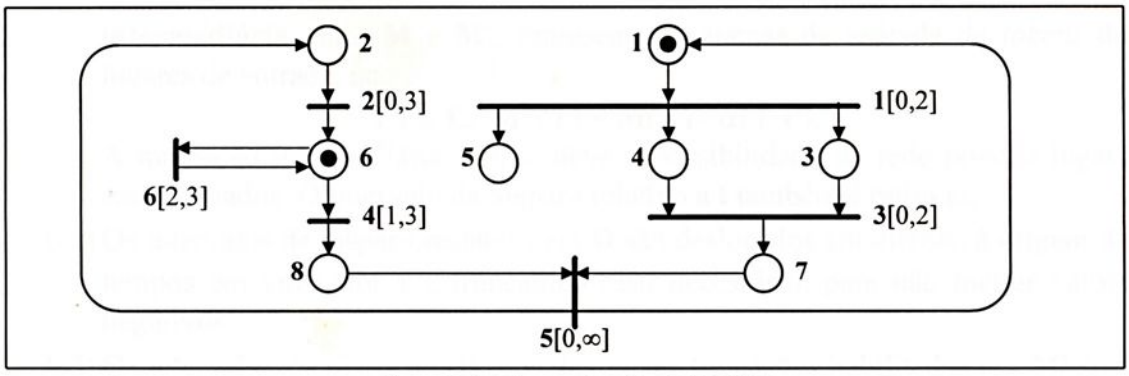


Fig. 3.1. Exemplo de RPTM.

A identificação de quais transições podem disparar numa RPTM é dada pela definição a seguir.

Definição 3.5: **Transição Disparável (em um estado).**

Numa RPTM, uma transição **t** é disparável no instante τ do estado **S** = (**M**,**D**) se as seguintes condições forem satisfeitas:

a) **t** está habilitada na marcação **M**, ou seja:

$$\forall \mathbf{l} \in \mathbf{L},\ \mathbf{M}(\mathbf{l}) \geq \alpha(\mathbf{l}, \mathbf{t});\ \text{e}$$

b) τ não é menor que o **TDI** da transição **t** e não é maior que o menor dos **TDF** de todas as demais transições habilitadas **k** pela marcação **M**, ou seja:

$$\mathbf{TDI}(\mathbf{t}) \leq \tau \leq \min_{\mathbf{k}} (\mathbf{TDF}(\mathbf{k})).$$

A condição estabelecida no item **a** da definição 3.5 é a regra de habilitação de uma transição para as RP L/T, conforme visto na definição 2.9 (definição equivalente).

A condição estabelecida no item **b** da definição 3.5 representa o fato de uma transição **t** habilitada **não poder disparar antes** de seu **TDI** dinâmico e **dever disparar até**, inclusive, o instante do menor dos **TDF's** dinâmicos, a menos que seja desabilitada pelo disparo de outra transição **conflitante k**. Duas transições **t** e **k** são conflitantes se possuem lugares de entrada comuns.

O tempo τ é contado a partir do momento da **geração** do estado **S**. Desta forma, o tempo absoluto, ou seja, aquele contado a partir do estado inicial $\mathbf{S_0}$, é igual a τ acrescido ao tempo para alcançar o estado **S**.

Definição 3.6: **Efeito do Disparo de uma Transição (em um estado).**

O efeito do disparo de uma transição disparável **t,** no instante τ do estado **S** = (**M**,**D**), é a passagem da rede para um estado **S'**= (**M'**,**D'**), de tal forma que:

a) A nova marcação **M'** é gerada como nas RP L/T (equivalente à definição 2.10):

$$\forall \mathbf{l} \in \mathbf{L},\ \mathbf{M'}(\mathbf{l}) = \mathbf{M}(\mathbf{l}) - \alpha(\mathbf{l}, \mathbf{t}) + \beta(\mathbf{l}, \mathbf{t});$$

b) O novo domínio **D'** é gerado em três passos:

b.1) Retira-se da expressão do domínio **D** os intervalos de disparo relativos às transições **k** que foram desabilitadas quando da retirada de *tokens* dos lugares de entrada promovida pelo disparo de **t**. Estas transições **k** são tais que **k** é habilitada em **M** e **k** não é habilitada em **M***, onde **M*** é a **marcação**

intermediária entre **M** e **M'**, representante apenas da retirada de *tokens* dos lugares de entrada, isto é:

$$\forall \mathbf{l} \in \mathbf{L},\ \mathbf{M^*(l) = M(l) - \alpha(l, t)}.$$

A marcação intermediária **M*** se deve à possibilidade da rede possuir lugares realimentados. O intervalo de disparo relativo a **t** também é retirado;

b.2) Os intervalos de disparo restantes em **D** são deslocados em direção à origem dos tempos em um valor τ e truncados, caso necessário, para não incluir valores negativos;

b.3) Os intervalos de disparo estáticos das novas transições habilitadas em **M'** (mas não habilitadas em **M***) são inseridos na nova expressão de domínio **D'**.

Notação: $\mathbf{S} \xrightarrow{(t,\tau)} \mathbf{S'}$

No exemplo da figura 3.1, a transição **1** pode disparar em qualquer instante no intervalo [0,2] e a transição **6** pode disparar apenas no instante 2 (o **TDI** da transição **6** é igual ao **TDF** da transição **1**, que é o menor dos **TDF**). Usando a definição 3.6, tem-se:

a) Disparando a transição **1** num tempo $\tau_1 \in [0,2]$, a rede passa para um estado $\mathbf{S_1} = (\mathbf{M_1}, \mathbf{D_1})$, onde:

$$\mathbf{M_1} = ((3,1),(4,1),(5,1),(6,1)) \text{ e}$$
$$\mathbf{D_1} = (\ 3[0,2],\ 4[1,3],\ 6[2-\tau_1, 3-\tau_1]\ ),\ 0 \le \tau_1 \le 2$$

Note que a transição **6** é remanescente quando a transição **1** dispara. Assim, seu intervalo foi deslocado em direção à origem dos tempos em um valor τ_1;

b) Disparando a transição **6** no instante 2, a rede passa para um estado $S_2 = (M_2, D_2)$, onde:

$$\mathbf{M_2} = ((1,1),(6,1)) \text{ e}$$
$$\mathbf{D_2} = (\ 1[0,0],\ 6[2,3]\ )$$

Os disparos das transições **3**, **4** e **6** no estado $\mathbf{S_1}$ e da transição **1** em $\mathbf{S_2}$ conduzem a rede para outros estados. Contudo, não estamos interessados na geração de estados, mas sim no que vamos caracterizar como classes de estados.

Definição 3.7: **Seqüência de Disparos Temporizados (σ,θ).**

Uma seqüência de disparos temporizados é o par (σ,θ), onde σ é uma seqüência de transições disparáveis e θ é uma seqüência de tempos.

$$(\sigma,\theta) = (\mathbf{t_1}, \tau_1), (\mathbf{t_2}, \tau_2), (\mathbf{t_3}, \tau_3),..., (\mathbf{t_n}, \tau_n)\ \text{, onde}$$
$$\mathbf{t_i} \in \mathbf{T} \text{ e } \tau_i \text{ é o instante de disparo de } \mathbf{t_i}\ ,\ 1 \le \mathbf{i} \le \mathbf{n}.$$

A seqüência de disparos temporizados (σ,θ) é responsável pela transição da rede de um estado **S** a um estado $\mathbf{S_n}$, se as transições $\mathbf{t_i}$ na seqüência σ são sucessivamente disparáveis em seus correspondentes instantes de tempo τ_i na seqüência θ.

Notação: $\mathbf{S} \xrightarrow{(\sigma,\theta)} \mathbf{S_n}$ ou

$$\mathbf{S} \xrightarrow{(t_1,\tau_1)} \mathbf{S_1} \xrightarrow{(t_2,\tau_2)} \mathbf{S_2} \cdots\cdots \mathbf{S_{n-1}} \xrightarrow{(t_n,\tau_n)} \mathbf{S_n}$$

onde $\mathbf{S_1}, \mathbf{S_2},..., \mathbf{S_{n-1}}$ são estados intermediários.

unesp

Definição 3.8: **Conjunto de Estados Decorrentes de S_0.**

É o conjunto de todos os estados pelos quais o sistema pode passar, partindo de um dado estado inicial S_0.

Notação: $\{S_0\}$.

O conjunto de estados decorrentes de S_0, assim como as correspondentes seqüências de disparos temporizados (σ,θ) **caracterizam** o comportamento da RPTM, tal qual a máquina de senhas para os demais tipos de RP. Entretanto, o fato de τ_i poder assumir uma infinidade de valores dentro do intervalo de disparo [**TDI**, **TDF**] de cada transição t_i habilitada numa marcação **M**, leva a rede a uma infinidade de estados.

Há um método [5,23] para a solução do problema de explosão de estados, o qual consiste na geração e enumeração de classes de estados, ou seja, agrupa-se o conjunto infinito de estados em um conjunto finito de classes.

3.5. Método de Enumeração de Classes de Estados para RPT de Merlin

O método de enumeração de classes de estados permite gerar uma representação do comportamento da RPTM. Tal comportamento será representado através de um grafo orientado onde os nós são as classes de estados e os arcos representam a relação de acesso entre as classes, a qual é provocada pelo disparo de uma transição. Este grafo é chamado de grafo de classes de estados.

Ao invés de considerar-se um dado estado **S** decorrente de S_0 através de uma seqüência de disparos temporizados (σ,θ), será considerado o conjunto dos estados gerados por todas as seqüências de disparos temporizados com a mesma seqüência σ e diferentes seqüências de tempo θ. Este conjunto de estados será chamado de classe de estados associados com a seqüência de disparos σ.

As classes são pares $\mathbf{C} = (\mathbf{M},\mathbf{D})$ onde:

- **M** é a **marcação da classe C**, ou seja, todas as classes em **C** têm a **mesma** marcação;
- **D** é o **domínio de disparo da classe**, definido como a **união** dos domínios $\mathbf{D_i}$ de todos os estados em **C**.

A figura 3.2 ilustra a transformação de estados para classes de estados. Nesta figura, S_{ni} representa o estado decorrente do estado inicial S_0 pela seqüência de disparos temporizados (σ,θ_i). A classe **C** é o conjunto de todos os estados S_{ni} ($i = 1, 2,...,\infty$).

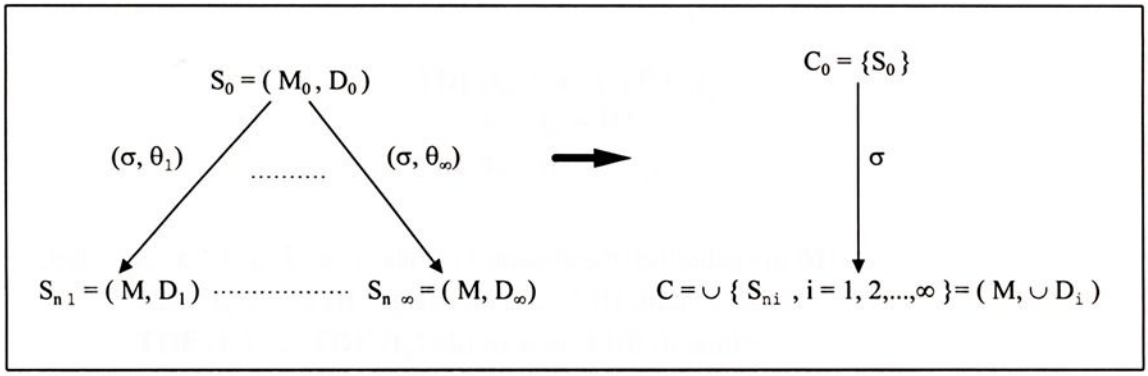


Fig. 3.2. Transformação de Estados em Classes de Estados.

Assume-se para a RPTM que a classe de estado atual da rede seja **C** = (**M**,**D**), onde **D** é o conjunto solução de um certo sistema de inequações lineares no qual a variável de tempo $\mathbf{x}_i$ é associada com a transição $\mathbf{t}_i$ habilitada na marcação **M**. Tem-se:

$$\mathbf{D} = \{ \mathbf{x} \mid \mathbf{A}.\mathbf{x} \geq \mathbf{b} \}$$

onde **A** é uma matriz, **b** é um vetor de constantes e **x** é um vetor cujas componentes são as variáveis $\mathbf{x}_i$.

As expressões de domínio para as classes de estados permitem introduzir **relações de interdependência** entre os tempos de disparo de diversas transições.

Definição 3.9: **Transição Disparável (em uma classe).**

Numa RPTM, uma transição **t** é disparável na classe **C** = (**M**,**D**) se as seguintes condições forem satisfeitas:

a) **t** está habilitada na marcação **M**, ou seja:

$$\forall \mathbf{l} \in \mathbf{L},\ \mathbf{M}(\mathbf{l}) \geq \alpha(\mathbf{l}, \mathbf{t});\ \text{e}$$

b) Assumindo que **t** é a **k**-ésima transição habilitada na marcação **M**, então o seguinte sistema de inequações é consistente:

$$\mathbf{A}.\mathbf{x} \geq \mathbf{b}$$
$$\mathbf{x}_k \leq \mathbf{x}_j\ ,\ \forall\ \mathbf{j},\ \mathbf{j} \neq \mathbf{k}$$

onde $\mathbf{x}_k$ e $\mathbf{x}_j$ representam a **k**-ésima e a **j**-ésima componentes de **x**.

Observa-se que a condição estabelecida pelo item 3.9.a é a regra de habilitação de uma transição para as RP L/T, conforme vimos na definição 2.9 (definição equivalente). Já a condição estabelecida pelo item 3.9.b significa que, no conjunto dos intervalos de tempos de disparos (intervalos denotados pelas variáveis $\mathbf{x}_i$) constituintes do domínio **D** da classe **C**, a transição $\mathbf{t}_k$ pode ser disparada antes ou no mesmo instante (de simulação) que qualquer outra transição habilitada $\mathbf{t}_j$ possa ser disparada.

Definição 3.10: **Efeito do Disparo de uma Transição (em uma classe).**

O efeito do disparo de uma transição disparável $\mathbf{t}_j$ na classe **C** = (**M**,**D**) é a passagem da rede para uma classe **C'**= (**M'**,**D'**), de tal forma que:

a) A nova marcação **M'** é gerada como nas RP L/T (equivalente à definição 2.10):

$$\forall \mathbf{l} \in \mathbf{L},\ \mathbf{M'}(\mathbf{l}) = \mathbf{M}(\mathbf{l}) - \alpha(\mathbf{l}, \mathbf{t}) + \beta(\mathbf{l}, \mathbf{t});$$

b) Seja o domínio **D** dado na seguinte forma:

$$\mathbf{TDI}(\mathbf{t}_i) \leq \mathbf{x}_i \leq \mathbf{TDF}(\mathbf{t}_i)$$
$$\mathbf{TDI}(\mathbf{t}_j) \leq \mathbf{x}_j \leq \mathbf{TDF}(\mathbf{t}_j)$$
$$\vdots$$
$$\mathbf{TDI}(\mathbf{t}_y) \leq \mathbf{x}_y \leq \mathbf{TDF}(\mathbf{t}_y)$$
$$\mathbf{x}_k - \mathbf{x}_n \leq \mathbf{H}^*_{kn}$$
$$\mathbf{x}_n - \mathbf{x}_k \leq \mathbf{H}^*_{nk}$$
$$\vdots$$

onde: - $\mathbf{t}_z$, **z** = **i**, **j**, **k**, **n**, **y** são as transições habilitadas em **M**;
- **TDI** ($\mathbf{t}_i$), ..., **TDI** ($\mathbf{t}_y$) são os seus **TDI** dinâmicos;
- **TDF** ($\mathbf{t}_i$), ..., **TDF** ($\mathbf{t}_y$) são os seus **TDF** dinâmicos;

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

- H^*_{kn} = [(**TDF** (t_k) em **D***) - (**TDI** (t_n) em **D***)], onde **D*** é o **domínio da classe C* que gerou a classe C**, ou seja, **C* é a classe antecessora de C**;
- H^*_{nk} = [(**TDF** (t_n) em **D***) - (**TDI** (t_k) em **D***)];
- Para a transição disparada t_j vale: **TDI** (t_j) $\leq$ min **TDF** (t_z).

O novo domínio **D'** é gerado em cinco passos [22], quais sejam:

b.1) Para cada transição t_k não desabilitada pelo disparo de t_j, o intervalo de disparo [**TDI**, **TDF**] em **D'** será dado por:

TDI (t_k) = máx {[0]; [$-H_{jk}$ se existir em **D**]; [(**TDI** (t_k) em **D**) - (menor **TDF** de todas as variáveis)]}

TDF (t_k) = min {[(**TDF** (t_k) em **D**) - (**TDI** (t_j) em **D**)]; [H_{kj} se existir em **D**]}

b.2) Para cada par de transições t_k e t_n não desabilitadas pelo disparo de t_j, tem-se:

$$x_k - x_n \leq H_{kn}$$

$$x_n - x_k \leq H_{nk}$$

onde: - H_{kn} = min {[(**TDF** (t_k) em **D**) - (**TDI** (t_n) em **D**)]; [H^*_{kn} se existir]};

- H_{nk} = min {[(**TDF** (t_n) em **D**) - (**TDI** (t_k) em **D**)]; [H^*_{nk} se existir]}.

b.3) As variáveis relativas às transições que foram desabilitadas pelo disparo de t_j não aparecerão em **D'**;

b.4) Também farão parte do domínio **D'** as variáveis correspondentes às transições que não eram habilitadas em **M** mas o são em **M'**. Cada uma dessas variáveis terá o seu intervalo de tempo determinado pelo **TDI** e **TDF** estáticos da transição correspondente;

b.5) Por fim, os **TDI** e os **TDF** dos intervalos das variáveis são subtraídos de um valor igual ao menor dos **TDI**.

A seguir, as regras dadas pela definição 3.10 serão usadas para gerar as classes de estados para a RPTM da figura 3.1.

A classe inicial é $C_0 = (M_0, D_0)$, onde:

M_0 = ((1,1),(6,1))

D_0: $0 \leq x_1 \leq 2$

$2 \leq x_6 \leq 3$

Da classe C_0, as transições **1** e **6** podem disparar, pois o menor dos **TDF** dinâmicos é 2 (**TDF** de x_1) e o **TDI** dinâmico da transição **6** é 2. Assim:

a) Disparando a transição **1** é gerada a classe $C_1 = (M_1, D_1)$, tal que:
- A marcação M_1 foi obtida da forma usual (definição 2.10);
- O domínio D_1 foi obtido usando as seguintes regras da definição 3.10:
  - 3.10.a: Novos valores para **TDI** (**6**) e **TDF** (**6**);
  - 3.10.c: Exclusão de x_1;
  - 3.10.d: x_3 e x_4 são as novas variáveis.

b) Disparando a transição **6** é gerada a classe $C_2 = (M_2, D_2)$, tal que:
- A marcação M_2 foi obtida da forma usual;
- O domínio D_2 foi obtido usando as seguintes regras:
  - 3.10.a: Novos valores para **TDI** (**1**) e **TDF** (**1**);
  - 3.10.d: x_6 é a nova variável.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Da classe $\mathbf{C_1}$, as transições **3**, **4** e **6** podem disparar. Lembrando que a marcação de uma classe é gerada de uma forma usual, tem-se que:

a) Disparando a transição **3** é gerada a classe $\mathbf{C_3} = (\mathbf{M_3}, \mathbf{D_3})$, tal que:
   - O domínio $\mathbf{D_3}$ foi obtido usando as seguintes regras:
     - 3.10.a: Novos valores para **TDI (4)**, **TDF (4)**, **TDI (6)** e **TDF (6)**;
     - 3.10.b: Inclusão das relações: $\mathbf{x_4} - \mathbf{x_6} \leq 3$
       $\mathbf{x_6} - \mathbf{x_4} \leq 2$;
     - 3.10.c: Exclusão de $\mathbf{x_3}$.

b) Disparando a transição **4** é gerada a classe $\mathbf{C_4} = (\mathbf{M_4}, \mathbf{D_4})$, tal que:
   - O domínio $\mathbf{D_4}$ foi obtido usando as seguintes regras:
     - 3.10.a: Novos valores para **TDI (3)** e **TDF (3)**;
     - 3.10.c: Exclusão de $\mathbf{x_4}$ e de $\mathbf{x_6}$.

c) Disparando a transição **6** é gerada a classe $\mathbf{C_5} = (\mathbf{M_5}, \mathbf{D_5})$, tal que:
   - O domínio $\mathbf{D_5}$ foi obtido usando as seguintes regras:
     - 3.10.a: Novos valores para **TDI (3)** e **TDF (3)**;
     - 3.10.c: Exclusão de $\mathbf{x_4}$ e $\mathbf{x_6}$;
     - 3.10.d: $\mathbf{x_4}$ e $\mathbf{x_6}$ são as novas variáveis.

De forma semelhante, obtém-se as demais classes (figura 3.3).

Classe	Marcação	Domínio
0	$M_0 = ((1,1),(6,1))$	$D_0: 0 \leq x_1 \leq 2$ $2 \leq x_6 \leq 3$
1	$M_1 = ((3,1),(4,1),(5,1),(6,1))$	$D_1: 0 \leq x_3 \leq 2$ $1 \leq x_4 \leq 3$ $0 \leq x_6 \leq 3$
2	$M_2 = ((1,1),(6,1))$	$D_2: 0 \leq x_1 \leq 0$ $2 \leq x_6 \leq 3$
3	$M_3 = ((5,1),(6,1),(7,1))$	$D_3: 0 \leq x_4 \leq 3$ $0 \leq x_6 \leq 3$ $x_4 - x_6 \leq 3$ $x_6 - x_4 \leq 2$
4	$M_4 = ((3,1),(4,1),(8,1))$	$D_4: 0 \leq x_3 \leq 1$
5	$M_5 = ((3,1),(4,1),(5,1),(6,1))$	$D_5: 0 \leq x_3 \leq 2$ $1 \leq x_4 \leq 3$ $2 \leq x_6 \leq 3$
6	$M_6 = ((7,1),(8,1))$	$D_6: 0 \leq x_5 \leq \infty$
7	$M_7 = ((5,1),(6,1),(7,1))$	$D_7: 1 \leq x_4 \leq 3$ $2 \leq x_6 \leq 3$
8	$M_8 = ((5,1),(6,1),(7,1))$	$D_8: 0 \leq x_4 \leq 3$ $0 \leq x_6 \leq 3$ $x_4 - x_6 \leq 1$ $x_6 - x_4 \leq 2$
9	$M_9 = ((3,1),(4,1),(5,1),(6,1))$	$D_9: 0 \leq x_3 \leq 0$ $1 \leq x_4 \leq 3$ $2 \leq x_6 \leq 3$
10	$M_{10} = ((1,1),(2,1))$	$D_{10}: 0 \leq x_1 \leq 2$ $0 \leq x_2 \leq 3$
11	$M_{11} = (2,1),(3,1),(4,1),(5,1))$	$D_{11}: 0 \leq x_2 \leq 3$ $0 \leq x_3 \leq 2$
12	$M_{12} = ((2,1),(5,1),(7,1))$	$D_{12}: 0 \leq x_2 \leq 3$

Fig. 3.3. Tabela de Classes de Estados para a RPTM da Figura 3.1.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

O grafo das classes de estados correspondente à RPTM da figura 3.1 é dado na figura 3.4.

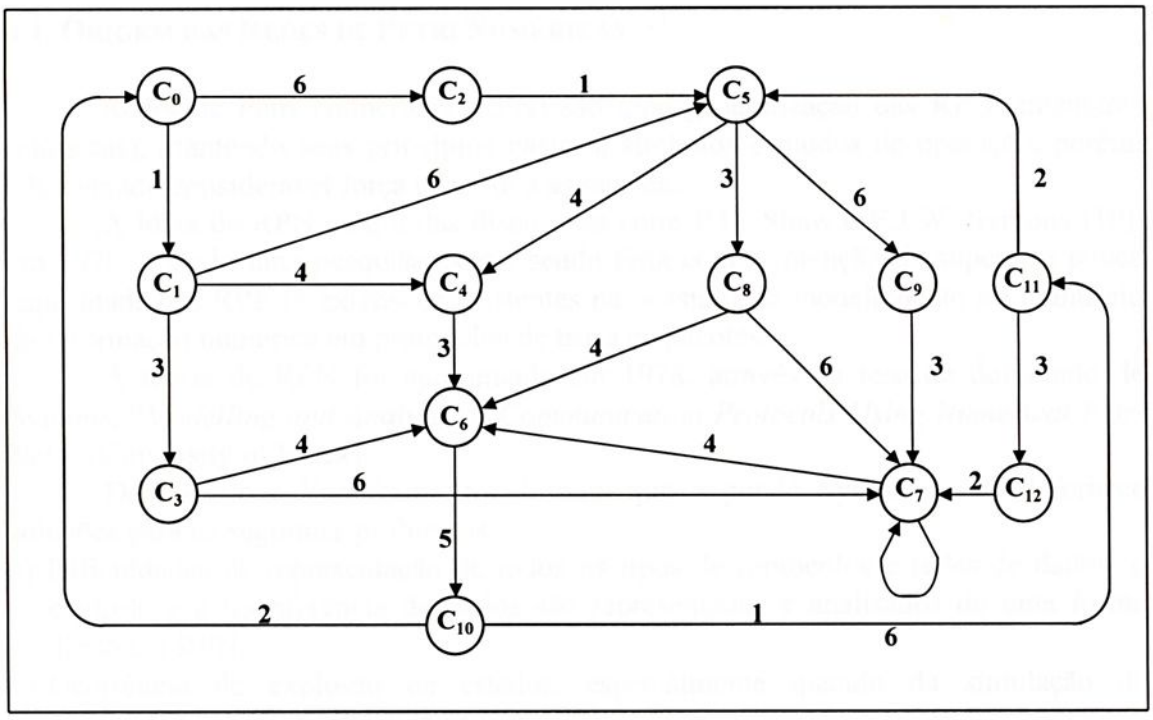


Fig. 3.4. Grafo das Classes de Estados para a RPTM da Figura 3.1.

O grafo das classes para as RPTM equivale à máquina de senhas das RP L/T. Propriedades de vivacidade, reiniciabilidade e limitabilidade podem ser definidas para RPTM de maneira análoga àquelas definidas para as RP L/T.

CAPÍTULO 4: REDES DE PETRI NUMÉRICAS

4.1. ORIGEM DAS REDES DE PETRI NUMÉRICAS

Redes de Petri Numéricas (RPN) são uma generalização das RP Elementares (clássicas), mantendo seus princípios básicos, símbolos e modos de operação, porém, adicionando considerável força descritiva às mesmas.

A idéia de RPN surgiu das discussões entre P.D. Shaw e F.J.W. Symons [10], em 1976, quando uma pesquisa estava sendo feita com a intenção de superar a pouca capacidade das RPE (e extensões existentes na ocasião) no modelamento do manuseio de informação numérica em protocolos de troca de pacotes.

A teoria de RPN foi apresentada em 1978, através da tese de doutorado de Symons, "*Modelling and Analysis of Communication Protocols Using Numerical Petri Nets*" (University of Essex).

Damasceno e Yamakami [6] afirmam que, segundo Symons, a RPN fornece soluções para os seguintes problemas:

a) Dificuldades de representação de todos os tipos de protocolos e redes de dados: o controle e a transferência de dados são representados e analisados de uma forma direta em RPN;
b) Ocorrência de explosão de estados, especialmente quando da simulação de protocolos de transferência de dados;
c) Dificuldades de formulação e declaração de situações complexas na análise da rede;
d) Dificuldades de integração dos resultados da análise e a verificação da validação dos protocolos: a utilização de RPN proporciona um considerável potencial para a integração da análise e verificação dos protocolos de comunicação tanto na fase de especificação, como nas fases de projeto e implementação.

Conforme pode-se observar pelo exposto acima, Symons adotou uma grande associação entre RPN e protocolos de comunicação. Entretanto, tal extensão de RP não se limita só a esta finalidade. De fato, pode-se representar um grande leque de sistemas utilizando RPN, devido à sua grande flexibilidade.

4.2. REGRAS DE FUNCIONAMENTO

A apresentação das Redes de Petri Numéricas será iniciada a partir de sua definição, após a qual será dado um conjunto de observações que facilitarão a compreensão da mesma.

***Definição 4.1:* Componentes de uma RPN.**

Uma RPN é definida por uma octupla $(\mathbf{L,T,H,E,S,O,P,E_0})$, onde:

- **L** é o conjunto de **lugares** da rede;
- **T** é o conjunto de **transições** da rede;
- **H** é o conjunto de **condições de habilitação** dos arcos de entrada;
- **E** é o conjunto de **regras de remoção de *tokens*** através dos arcos de entrada;

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

- **S** é o conjunto de **regras de inserção de *tokens*** através dos arcos de saída;
- **O** é o conjunto de **operações das transições**;
- **P** é o conjunto de **predicados das transições**; e
- $\mathbf{E_0}$ é o **estado inicial** da RPN, denotado por $\mathbf{E_0} = (\mathbf{M_0},\mathbf{V_0})$, onde:
  - $\mathbf{M_0}$ é a **marcação inicial**;
  - $\mathbf{V_0}$ é o conjunto de **valores iniciais para as variáveis** da rede.

Observações:

a) *Tokens* podem ter **qualquer natureza** e valor. Assim, não precisam necessariamente ser representados pelo símbolo ●. Podem ser representados por símbolos diferentes, conforme o sistema modelado;
b) Uma **memória de escrita e leitura** é associada à rede. Tal memória é representada por um conjunto de variáveis;
c) O conceito de marcação é ampliado para o de **estado**. O estado de uma RPN é dado por $\mathbf{E_i} = (\mathbf{M_i}, \mathbf{V_i})$, onde $\mathbf{M_i}$ é uma **marcação** denotando quantos ou quais os *tokens* presentes nos lugares da rede e $\mathbf{V_i}$ é um conjunto de **variáveis** e seus respectivos valores;
d) As regras de habilitação de uma transição, além de envolverem os lugares de entrada da mesma, envolvem também um **predicado** associado à transição. A **condição de habilitação** de cada arco de entrada e o predicado da transição devem ser verdadeiros para que a mesma esteja habilitada;
e) O **disparo** de uma transição promove a remoção de *tokens* de seus lugares de entrada (de acordo com a regra de **remoção de *tokens*** de cada arco de entrada da transição), a **adição de *tokens*** em seus lugares de saída (de acordo com a regra de inserção de *tokens* de cada arco de saída da transição) e a **realização de operações** sobre a **memória** da rede (conjunto de operações da transição).

A figura 4.1 ilustra os componentes de uma RPN.

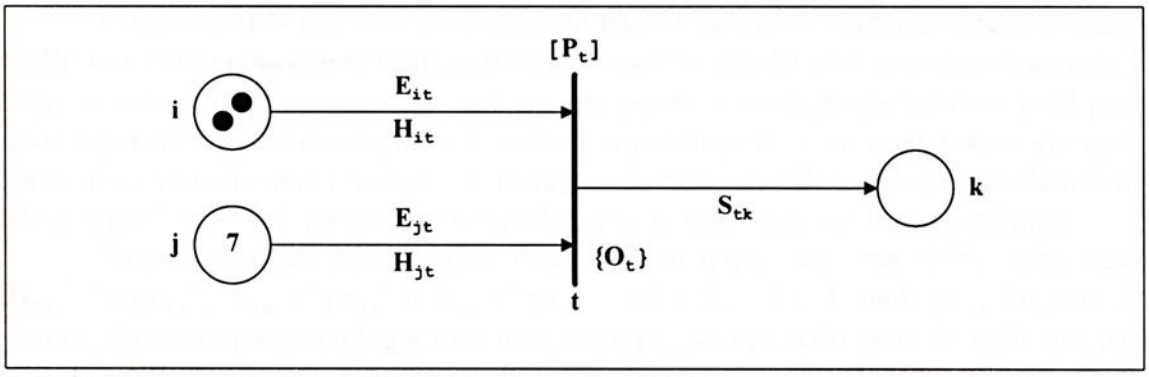


Fig. 4.1. Componentes de uma RPN.

Adota-se as seguintes convenções gráficas para a representação de uma RPN:

a) Lugares e transições são nomeados com números inteiros, por exemplo, **i**,**j**,**k**,**t**;
b) Quando não importar sua **capacidade de distinção**, *tokens* são representados por ●, caso contrário, por números inteiros;
c) O predicado de uma transição deve ser anotado entre colchetes;
d) O conjunto de operações de uma transição deve ser anotado entre chaves;

e) A regra de inserção de *tokens* de uma transição **t** para um lugar de saída **k** ($\mathbf{S_{tk}}$) pode ser anotada em qualquer região próxima ao arco correspondente;
f) Considerando o centro de uma transição **t** como sendo o centro de um relógio, e caminhando no sentido horário, deve-se anotar:
   f.1) antes de cruzar o arco correspondente, a condição de habilitação do lugar de entrada **i** de **t** ($\mathbf{H_{it}}$);
   f.2) após cruzar o arco correspondente, a regra de remoção de *tokens* do lugar de entrada **i** de **t** ($\mathbf{E_{it}}$).

Formalizando, tem-se as regras de funcionamento de RPN dadas a seguir.

Definição 4.2: **Regra de Habilitação.**

Uma transição **t** está habilitada perante um dado estado $\mathbf{E_i} = (\mathbf{M_i},\mathbf{V_i})$ se as seguintes condições forem satisfeitas:

a) A condição de habilitação do lugar de entrada **i** de **t** , $\mathbf{H_{it}}$, for verdadeira para todo lugar de entrada **i** de **t**;
b) O predicado da transição **t** , $\mathbf{P_t}$, for verdadeiro.

Definição 4.3: **Efeito do Disparo de uma Transição.**

Quando uma transição habilitada **t** dispara, as seguintes ações são realizadas em seqüência, como uma operação indivisível:

a) *Tokens* são removidos dos lugares de entrada **i** de **t**, de acordo com as regras de remoção de *tokens* $\mathbf{E_{it}}$, $\forall$ **i** | **i** é lugar de entrada de **t**;
b) A operação da transição **t**, $\mathbf{O_t}$, é efetuada sobre a memória da rede, ou seja, é gerada uma nova **valoração** para o conjunto de variáveis da rede;
c) *Tokens* são inseridos nos lugares de saída **k** de **t**, de acordo com as regras de inserção de *tokens* $\mathbf{S_{tk}}$, $\forall$ **k** | **k** é lugar de saída de **t**.

Como as RPN são uma generalização das RP originais é interessante considerar as RP L/T como uma classe restrita de RPN. Nas RPE não há uma memória associada à rede, os *tokens* são representados unicamente por ●, a condição de habilitação **H** para cada lugar de entrada é sempre "≥1" (isto é, a presença de 1 ou mais *tokens* em cada lugar de entrada de uma transição) e as regras de remoção (**E**) e inserção (**S**) de *tokens* são sempre "1" (isto é, apenas 1 *token* pode ser removido ou inserido em um lugar).

Para representar uma rede de Petri L/T em termos de uma RPN, basta fazer $\mathbf{H_{lt}}$ = "≥ $\mathbf{pe_{lt}}$", $\mathbf{E_{lt}}$ = "$\mathbf{pe_{lt}}$" e $\mathbf{S_{tl}}$ = "$\mathbf{ps_{tl}}$" , $\forall$ **l** ∈ **L**, $\forall$ **t** ∈ **T**, onde $\mathbf{pe_{lt}}$ é o peso de entrada do ramo que une o lugar **l** à transição **t** e $\mathbf{ps_{tl}}$ é o peso do ramo de saída que une a transição **t** ao lugar **l**. Entretanto, não é de interesse representar uma rede L/T em termos de uma RPN. A capacidade de modelagem desta é muito superior à da primeira.

A figura 4.2.a mostra uma rede elementar, enquanto 4.2.b mostra a sua RPN correspondente.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

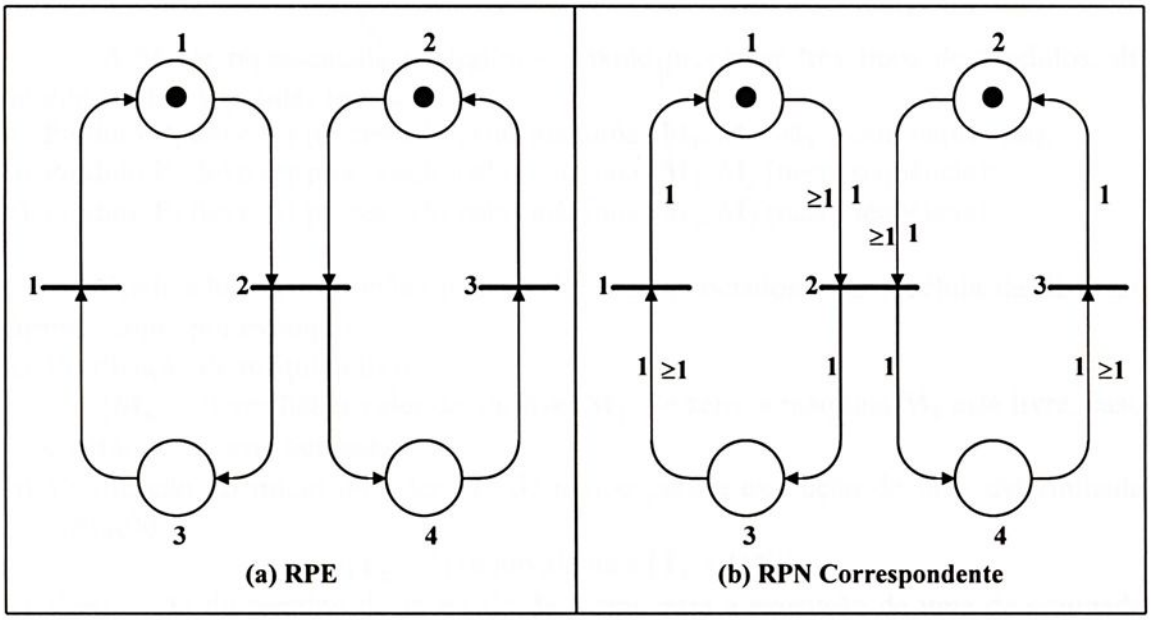


Fig. 4.2. Exemplo de RPE e sua RPN Correspondente.

Na seção seguinte, será dado um exemplo de aplicação prática para RPN, associada à RPT de Ramchandani.

4.3. Modelo RPN de uma Célula Flexível de Manufatura

Nesta seção, mostrar-se-á como RPN podem ser usadas para a representação de uma célula flexível de manufatura.

Inicialmente, será visto como associar predicados e operações de transições ao conjunto de regras de uma célula. Exemplo de uma célula é dado na figura 4.3, conforme apresentado em [6], a qual é composta por três máquinas, um robô e dois acessos (um de entrada e outro de saída).

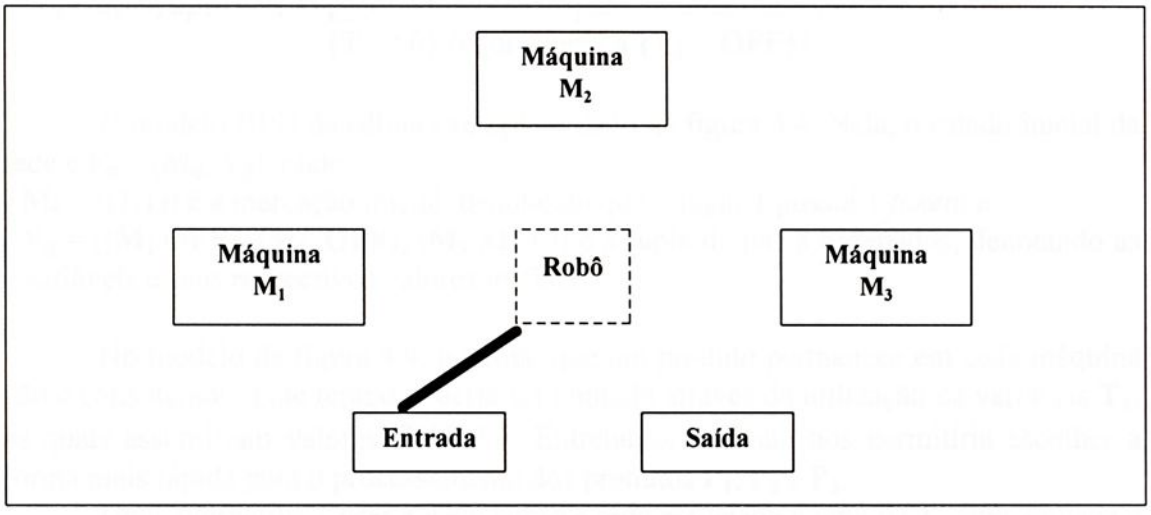


Fig. 4.3. Célula Flexível de Manufatura.

A célula representada na figura 4.3 pode processar três tipos de produtos, de acordo com as seguintes regras:

a) Produto $\mathbf{P_1}$ deve ser processado pelas máquinas $\mathbf{M_1}$, $\mathbf{M_2}$, $\mathbf{M_3}$ (nesta seqüência);
b) Produto $\mathbf{P_2}$ deve ser processado pelas máquinas $\mathbf{M_2}$, $\mathbf{M_3}$ (nesta seqüência);
c) Produto $\mathbf{P_3}$ deve ser processado pelas máquinas $\mathbf{M_1}$, $\mathbf{M_3}$ (nesta seqüência).

Predicados de transições RPN podem ser associados a uma célula de diversas formas, como por exemplo:

a) Verificação de máquina livre:
   $[\mathbf{M_i} = 0]$ verifica o valor da variável $\mathbf{M_i}$. Se zero, a máquina $\mathbf{M_i}$ está livre, caso contrário, $\mathbf{M_i}$ está ocupada;
b) Verificação do início do intervalo de tempo para a execução de uma determinada operação:
   $[\mathbf{T_i} = 1]$ (equivalente a $[\mathbf{T_i} = \mathbf{ON}]$);
c) Verificação do término do intervalo de tempo para a execução de uma determinada operação:
   $[\mathbf{T_i} = 0]$ (equivalente a $[\mathbf{T_i} = \mathbf{OFF}]$),
   onde $\mathbf{M_i}$ e $\mathbf{T_i}$ são variáveis do conjunto de variáveis **V** da rede.

Por outro lado, as operações das transições são as responsáveis por **atribuir** valores às variáveis da rede, o que pode significar:

a) A liberação de uma máquina após a ação promovida pelo disparo de uma transição:
   $\{\mathbf{M_i} = 0\}$ (equivalente a $\{\mathbf{M_i} = \mathbf{OFF}\}$);
b) A ocupação de uma máquina após a ação promovida pelo disparo de uma transição:
   $\{\mathbf{M_i} = 1\}$ (equivalente a $\{\mathbf{M_i} = \mathbf{ON}\}$);
c) O início do intervalo de tempo para a execução de uma determinada operação, após a ação promovida pelo disparo de uma transição:
   $\{\mathbf{T_i} = 1\}$ (equivalente a $\{\mathbf{T_i} = \mathbf{ON}\}$);
d) O encerramento do intervalo de tempo para a execução de uma determinada operação, após a ação promovida pelo disparo de uma transição:
   $\{\mathbf{T_i} = 0\}$ (equivalente a $\{\mathbf{T_i} = \mathbf{OFF}\}$).

O modelo RPN da célula exemplo é dado na figura 4.4. Nela, o estado inicial da rede é $\mathbf{E_0} = (\mathbf{M_0}, \mathbf{V_0})$, onde:

- $\mathbf{M_0} = ((1,1))$ é a marcação inicial, denotando que o lugar **1** possui 1 *token*; e
- $\mathbf{V_0} = ((\mathbf{M_1},\mathbf{OFF}), (\mathbf{M_2},\mathbf{OFF}), (\mathbf{M_3},\mathbf{OFF}))$ é a tupla de pares ordenados, denotando as variáveis e seus respectivos valores iniciais.

No modelo da figura 4.4, o tempo que um produto permanece em cada máquina não é considerado. Este tempo poderia ser contado através da utilização de variáveis $\mathbf{T_i}$, as quais assumiriam valores **ON/OFF**. Entretanto, isto não nos permitiria escolher a forma mais rápida para o processamento dos produtos $\mathbf{P_1}$, $\mathbf{P_2}$ e $\mathbf{P_3}$.

Uma alternativa é usar a RPN como a rede associada a uma RPT. Será adotada a RPT de Ramchandani (seção 3.2.2), uma vez que a mesma oferece uma forma mais direta para a resolução do problema proposto. Será associado um valor de tempo a cada

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

transição, indicando o tempo gasto em cada máquina na parte que lhe cabe no processamento de um produto.

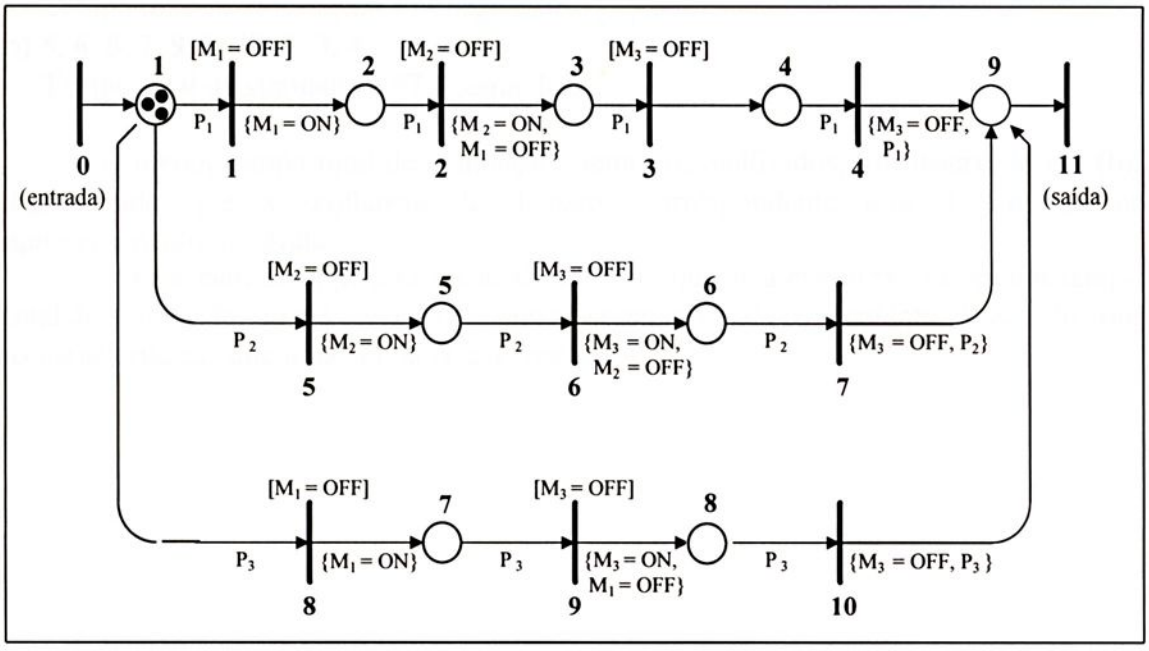


Fig. 4.4. RPN Modelando a Célula Flexível de Manufatura da Figura 5.3.

Atribuindo os tempos de processamento de cada máquina para cada um dos produtos, pode-se determinar o melhor desempenho da célula através da determinação, por meio da simulação da rede, da seqüência de disparos de transições para a qual o tempo de processamento é o menor.

A rede da figura 4.5 apresenta uma RPN associada a uma RPT de Ramchandani, para modelar também os aspectos temporais da célula de manufatura.

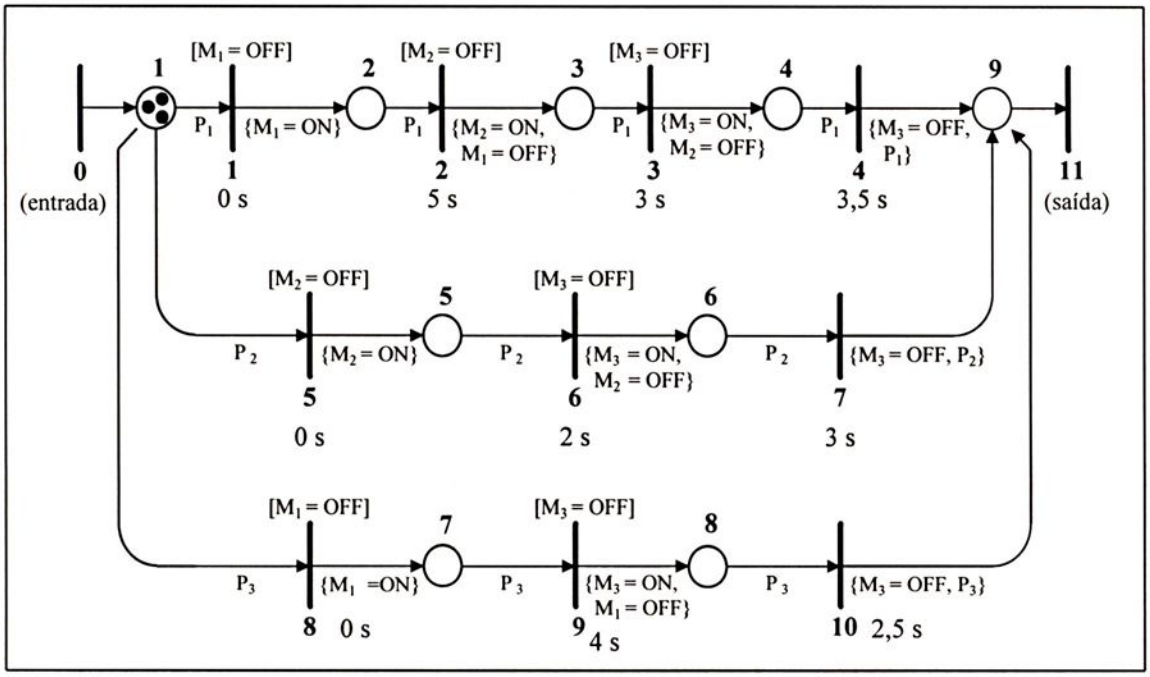


Fig. 4.5. Modelo da Célula Flexível de Manufatura em RPN/RPT de Ramchandani.

Para exemplificar, a referência [6] considera as duas seqüências de disparos:

a) **1, 2, 8, 3, 5, 4, 9, 10, 6, 7**
   Tempo total de simulação: 114 segundos;
b) **5, 6, 8, 7, 9, 1, 10, 2, 3, 4**
   Tempo total de simulação: 97.5 segundos.

O menor tempo total de simulação, dentre os analisados, foi observado em (b), significando que a seqüência de disparos correspondente leva a um melhor aproveitamento da célula.

Outros caminhos podem ser analisados até que seja encontrado o menor tempo total de simulação, ou até que seja encontrado o tempo mais conveniente, de acordo com as características que uma célula possa apresentar.

unesp

CAPÍTULO 5: REDES DE PETRI ESTOCÁSTICAS

5.1. PRINCIPAIS CLASSES DE REDES DE PETRI ESTOCÁSTICAS

A idéia por trás da modelagem em Rede de Petri Estocástica é representar o sistema a ser analisado por uma RP, sobre a qual aplicar-se-ão os **métodos da teoria de processos estocásticos**.

A seguir, é apresentada uma visão geral das principais classes de Redes de Petri Estocásticas, quais sejam:

a) Redes de Petri Estocásticas de *Florin* e *Natkin* (RPE-FN) [11];
b) Redes de Petri Estocásticas de *Molloy* (RPE-M) [23];
c) Redes de Petri Estocásticas Generalizadas de *Marsan*, *Balbo* e *Conte* (RPEG) [21]; e
d) Redes de Petri Estocásticas Extendidas de *Dugan* (RPEE) [13].

Estas classes de redes são definidas a partir de uma Rede de Petri Lugar/Transição (capítulo 2) e diferem, basicamente, na forma como elas alteram as regras de habilitação e de disparo daquela classe de rede. As mudanças introduzidas são dadas a seguir [2].

5.2. MUDANÇAS INTRODUZIDAS PELAS REDES DE PETRI ESTOCÁSTICAS

5.2.1. ASSOCIAÇÃO DE UM TEMPO DE DISPARO ALEATÓRIO ÀS TRANSIÇÕES

O tempo entre o instante no qual uma transição torna-se habilitada e o instante no qual ela dispara é considerado como uma variável aleatória contínua $\mathbf{X_t}$, tomada de valores reais não-negativos, à qual é associada a função de distribuição:

$$\mathbf{F_t(x) = P\{X_t \leq x\}}.$$

Isto é feito substancialmente de três formas:

a) ***Associação de uma função de distribuição exponencial a cada transição t:***

$$\mathbf{F_t = 1 - e^{-\lambda_t x}}$$

onde $\lambda_t > 0$ é o parâmetro real e **x** é um real não-nulo.

Pode-se mostrar que:

a.1) A probabilidade de duas transições serem disparadas no mesmo instante é zero;
a.2) A máquina de senhas (grafo de alcançabilidade) da rede estocástica é isomorfa a uma **cadeia de Markov homogênea** e, como tal, pode ser encontrada.

Em relação a variáveis aleatórias contínuas, somente as funções de distribuição exponenciais satisfazem a propriedade de *Markov*. Assim, o fato de uma função de distribuição exponencial ser associada a cada transição é uma condição necessária e suficiente para a aplicação dos métodos da teoria de cadeias de *Markov* à máquina de senhas da rede estocástica.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

b) ***Associação de funções de distribuição arbitrárias às transições, exceto para:***

b.1) Todas as transições concorrentemente habilitadas em alguma marcação alcançável, as quais devem ser distribuídas exponencialmente; e

b.2) Todas as transições em conflito em alguma marcação alcançável, as quais devem ser distribuídas de tal forma que, se uma delas tornar-se desabilitada, a distribuição de probabilidade da mesma em uma habilitação subseqüente seja independente do tempo de disparo anterior.

Desta forma, a máquina de senhas da rede estocástica pode ser considerada como um **semi processo de Markov** e, como tal, pode ser encontrada.

c) ***Associação de funções de distribuição completamente arbitrárias às transições.***

Neste caso, como inexiste uma **cadeia de Markov** isomorfa à máquina de senhas, tem-se que recorrer à simulação para encontrar esta última.

5.2.2. Transições Imediatas

As transições imediatas disparam tão logo tornam-se habilitadas, sem qualquer consumo de tempo. Elas podem ser definidas explicitamente ou implicitamente, bastando permitir ao parâmetro λ_t de uma transição **t** distribuída exponencialmente assumir o valor $+\infty$.

Graficamente, **transições imediatas** são geralmente representadas por uma **barra fina**, enquanto **transições temporizadas** são representadas por uma **barra grossa**, conforme a figura 5.1.

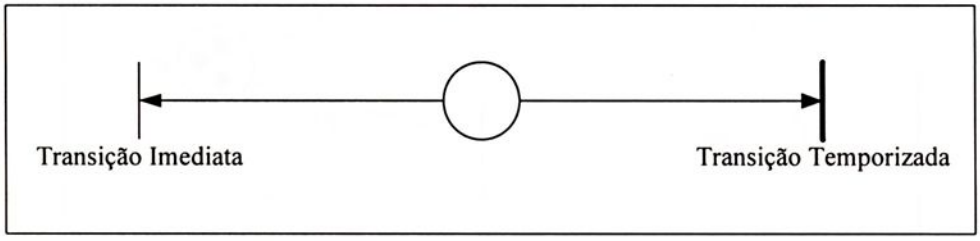


Fig. 5.1. Representação de Transições Imediata e Temporizada.

O uso de transições imediatas é problemático no caso de conflitos. Alguns autores exigem, por convenção, que:

a) Conflitos entre transições imediatas sejam proibidos, por serem contraditórios;

b) Conflitos entre uma transição imediata e uma temporizada sejam resolvidos em favor da primeira.

5.2.3. Arcos Probabilísticos / Chaves Aleatórias

Um **arco probabilístico** é um arco com duas pontas, saindo de uma transição para dois lugares, conforme figura 5.2. Nela, o efeito do disparo de **t** produz, com probabilidade **a**, um *token* em **p** e nenhum em **q**, ou, com probabilidade **1-a**, um *token* em **q** e nenhum em **p**.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

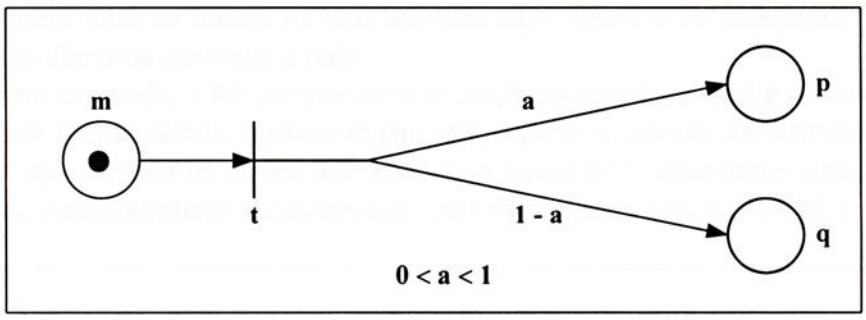


Fig. 5.2. Arco Probabilístico.

Uma **chave aleatória** é uma distribuição de probabilidade associada à escolha da transição a ser disparada dentre um conjunto de transições imediatas, conflitantes em uma dada marcação. Esta distribuição é geralmente **dependente de marcações**. Desta forma, a dificuldade de usar tais chaves é que as mesmas só podem ser definidas durante a construção da máquina de senhas.

O uso de chaves aleatórias acaba com a noção de matriz de incidência e, com isso, acaba também com a noção de invariantes de lugares e de invariantes de transições (os quais constituem um método de análise de RP), mesmo que a chave aleatória não seja dependente de marcação. Consideremos, por exemplo, a rede da figura 5.3, a qual possui uma chave aleatória independente de marcação (a probabilidade de disparo de cada transição está escrita nos arcos de entrada correspondentes).

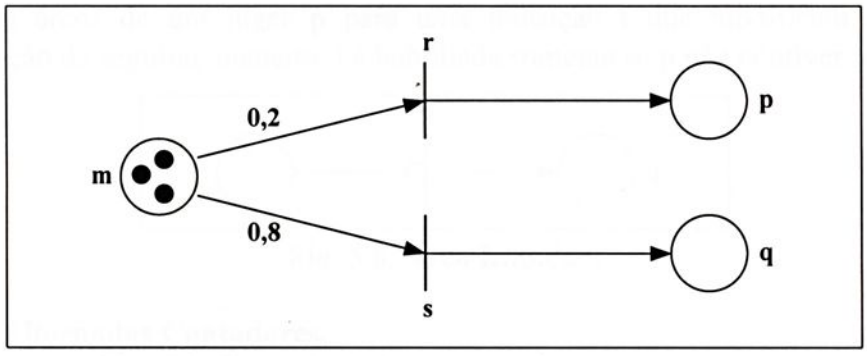


Fig. 5.3. Exemplo de Rede com Chaves Aleatórias.

Conforme a definição 2.12 do capítulo 2, uma "matriz de incidência" para a rede da figura 5.3 é apresentada na figura 5.4.

$$C = \begin{array}{c} \begin{matrix} m & p & q \end{matrix} \\ \begin{bmatrix} -0{,}2 & 1 & 0 \\ -0{,}8 & 0 & 1 \end{bmatrix} \end{array} \begin{matrix} r \\ s \end{matrix}$$

Fig. 5.4. Pseudo Matriz de Incidência da Rede da Figura 5.3.

Pode-se ver, analisando a matriz **C** pela teoria de invariantes de lugar (S-invariantes), que não há S-invariantes na rede da figura 5.3, o que contradiz com o

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

fato do número total de *tokens* na rede ser constante (igual a 3), independentemente da seqüência de disparos atribuída à rede.

Como esperado, a RP subjacente tem um S-invariante, o qual é calculado a partir de sua matriz de incidência verdadeira (ou seja, aquela resultante da retirada das chaves aleatórias), apresentada na figura 5.5. Usando a teoria de S-invariantes obtém-se, como S-invariante correspondente à conservação de *tokens* observada, o vetor $[1\ 1\ 1]^T$.

$$C = \begin{array}{c} \begin{matrix} \mathbf{m} & \mathbf{p} & \mathbf{q} \end{matrix} \\ \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \end{array} \begin{matrix} \mathbf{r} \\ \mathbf{s} \end{matrix}$$

Fig. 5.5. Matriz de Incidência da RP Subjacente à Rede da Figura 5.3.

5.2.4. Arcos Anormais

Há dois tipos de **arcos anormais**, os quais vão contra a interpretação usual de RP e, ainda pior, tornam inaplicáveis a maior parte dos conceitos e resultados de sua teoria.

a) **Arcos Inibidores.**

São arcos de um lugar **p** para uma transição **t** que modificam a regra de habilitação da seguinte maneira: **t** é habilitada somente se **p** não contiver *tokens*.

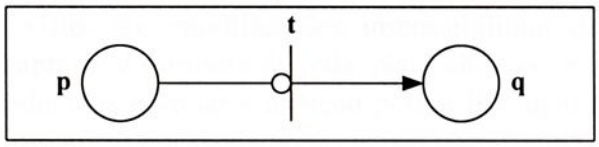


Fig. 5.6. Arco Inibidor.

b) **Arcos Alternados Contadores.**

Seja um arco com peso **k** indo de um lugar **p** para uma transição **t**. Um **arco alternado contador** é um arco indo de um lugar **p** para uma transição **s**, o qual altera a regra de habilitação de **s** da seguinte maneira: **s** pode disparar uma vez para cada *token* presente em **p** se o número de *tokens* em **p** for menor do que **k**. Ademais, o disparo de **s** não altera a marcação de **p**.

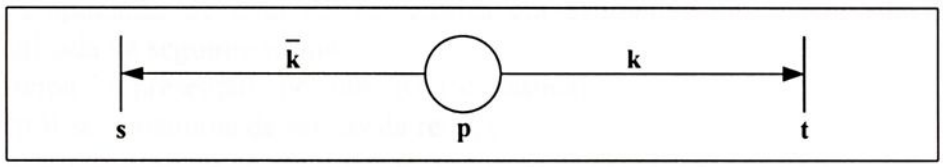


Fig. 5.7. Arco Alternado Contador.

Apresentadas as características das principais classes de RP estocásticas, a figura 5.8 sintetiza o relacionamento estrutural entre as mesmas.

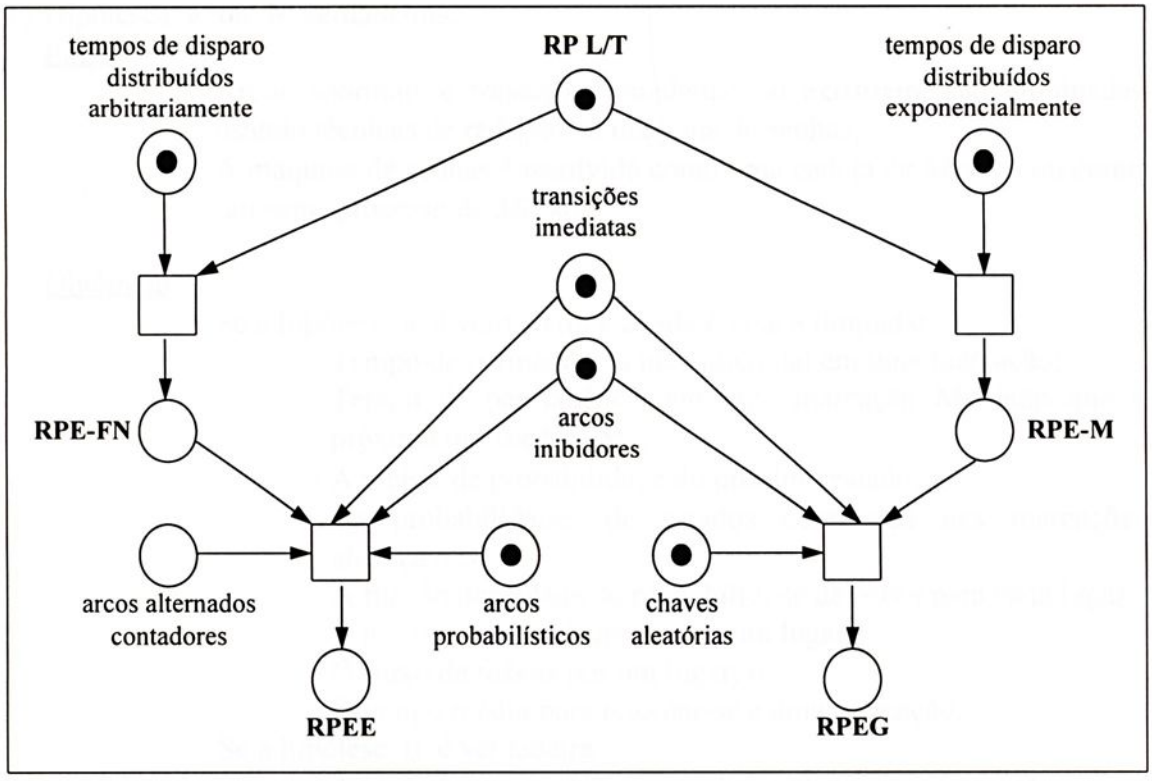


Fig. 5.8. Relacionamento Estrutural entre os Principais Tipos de RP Estocásticas.

Fora estes quatro tipos, existem muitos outros tipos de redes estocásticas, cada qual com seus próprios conceitos e modificações das regras de habilitação e de disparo. Conforme pode ser visto, tais modificações impossibilitam definir uma matriz de incidência capaz de capturar a estrutura da rede. Naturalmente, é possível desconsiderar as modificações introduzidas e voltar a atenção para a RP lugar/transição subjacente à estocástica, na qual tornam-se possíveis análises tais como as de invariantes de lugar e de transição. Entretanto, isto restringe as redes estocásticas a uma melhor ilustração de alguns métodos de cálculo da teoria dos processos estocásticos.

Além disso, o significado dos resultados transferidos diretamente das RP subjacentes às estocásticas ainda é uma questão aberta.

5.3. Estimativas de Desempenho

A aplicação de uma RP estocástica em estimativa de desempenho pode ser esquematizada da seguinte forma:

1) O sistema é representado por uma RP estocástica;
2) Constrói-se a máquina de senhas da rede; e
3) Associam-se funções de distribuição de tempo de disparo às transições.

A partir disso, tem-se duas possibilidades (considerando-se as três formas de associação de tempos de disparo às transições - itens **a**, **b** e **c** da seção 5.2.1):

unesp

1) Hipóteses **a** ou **b** verdadeiras.
Então:

- Arcos anormais e transições imediatas, se existirem, são eliminadas usando técnicas de redução na máquina de senhas;
- A máquina de senhas é resolvida como uma cadeia de *Markov* ou como um semi-processo de *Markov*.

Obtém-se:

- Se a hipótese **a** é verdadeira e a rede é viva e limitada:
  - Tempo de permanência incondicional em uma marcação;
  - Tempo de permanência em uma marcação $\mathbf{M_i}$ dado que a próxima marcação é $\mathbf{M_j}$;
  - A matriz de probabilidade do próximo estado;
  - As probabilidades de estados constantes nas marcações alcançáveis;
  - A função densidade de probabilidade de *token* para cada lugar;
  - O número médio de *tokens* em um lugar;
  - O fluxo de *tokens* por um lugar; e
  - O tempo médio para retornar-se a uma marcação.
- Se a hipótese **b** é verdadeira:
  - Tempo de permanência incondicional em uma marcação;
  - Tempo de permanência em uma marcação $\mathbf{M_i}$ dado que a próxima marcação é $\mathbf{M_j}$;
  - A distribuição do tempo de permanência em uma marcação $\mathbf{M_i}$ dado que a próxima marcação é $\mathbf{M_j}$;
  - A matriz de probabilidade do próximo estado.

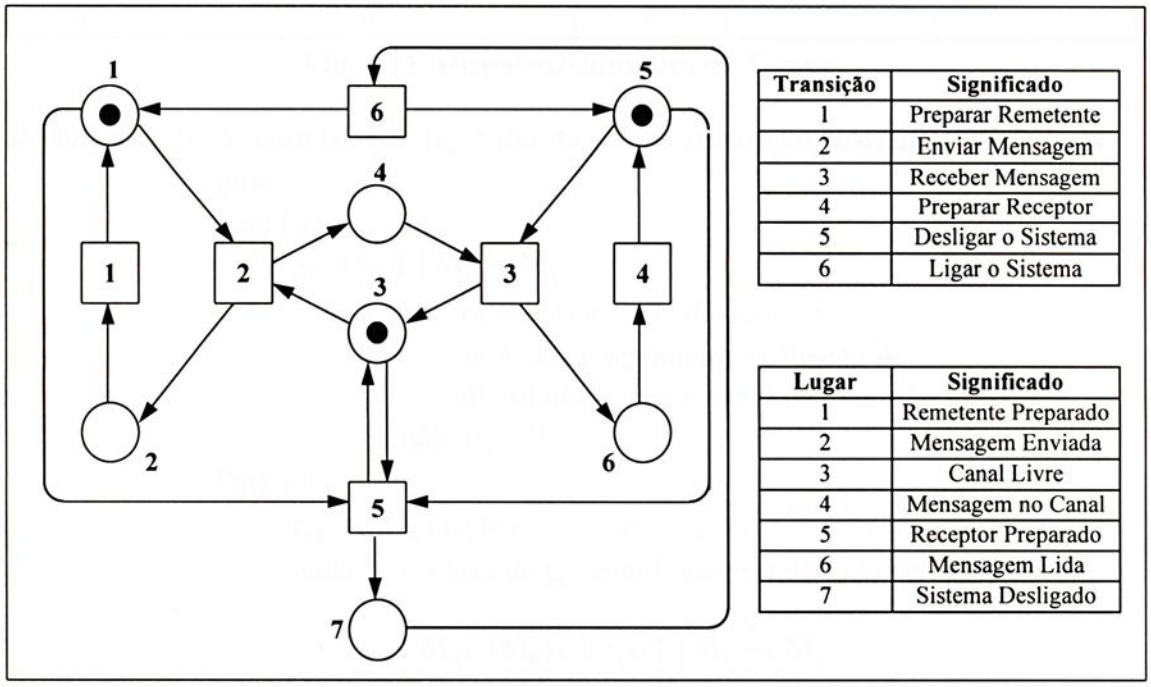
2) Hipótese **c** verdadeira.
Então:

- Simular a rede, o que pode ser feito de duas maneiras:
  - Simulação Breve, para a obtenção de valores tais como probabilidades de saída e distribuição de tempo de saída;
  - Simulação Ergódica, para obter termos longos ou medidas de média.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

5.4. Procedimentos de Cálculo de Estimativas de Desempenho

Considere a RP lugar/transição de um sistema emissor-receptor (figura 5.9).



Transição	Significado
1	Preparar Remetente
2	Enviar Mensagem
3	Receber Mensagem
4	Preparar Receptor
5	Desligar o Sistema
6	Ligar o Sistema

Lugar	Significado
1	Remetente Preparado
2	Mensagem Enviada
3	Canal Livre
4	Mensagem no Canal
5	Receptor Preparado
6	Mensagem Lida
7	Sistema Desligado

Fig. 5.9. RP Lugar/Transição de um Sistema Emissor-Receptor.

A figura 5.10 apresenta a máquina de senhas correspondente a rede dada.

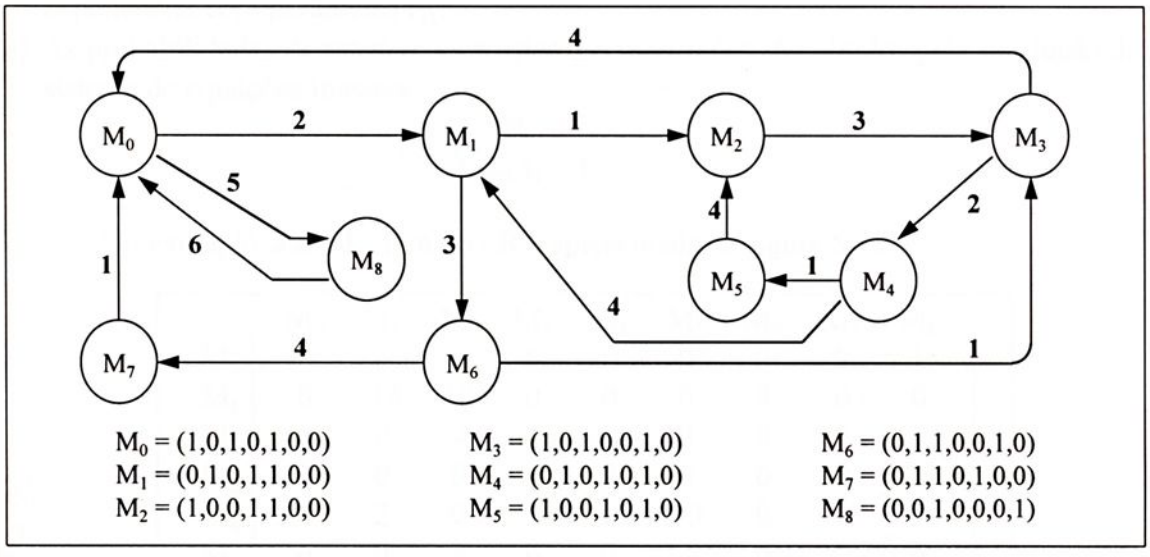


Fig. 5.10. Máquina de Senhas da Rede da Figura 5.9.

Deve-se associar uma função de distribuição exponencial de tempo de disparo a cada transição. Considere, por exemplo, as taxas apresentadas na figura 5.11.

Transição	Taxa (Disparos/Unidade de Tempo)	Transição	Taxa (Disparos/Unidade de Tempo)
1	10	4	2
2	4	5	1
3	4	6	1

Fig. 5.11. Disparos/Unidades de Tempo.

Definição 5.1: **A matriz $R = (r_{i,j})$ das taxas de transição instantâneas é dada por:**

Para $i \neq j$:

Se $\exists\, t \in T \mid M_i \xrightarrow{t} M_j$

Então $r_{ij} = d(1 - e^{-\lambda \tau}) / d\tau_{\mid \tau = 0} = \lambda$

onde λ é o parâmetro da função de distribuição exponencial da transição **t**.

Senão $r_{ij} = 0$.

Para $i = j$:

$$r_{ij} = d \prod_k (1-(1 - e^{-\lambda_k \tau}) / d\tau_{\mid \tau = 0} = d\, e^{-\tau \sum_k \lambda_k} / d\tau_{\mid \tau = 0} = -\sum_k \lambda_k$$

onde λ_k é a taxa de t_k, com **k** assumindo todo valor $\neq i$ tal que

$$\exists\, M_k \in \{M_0\} \text{ e } \exists\, t_k \in T \mid M_i \xrightarrow{t_K} M_k .$$

Desta forma, tem-se as seguintes observações:

a) O tempo de espera incondicional em M_i é exponencial com parâmetro $-r_{ij}$;

b) O tempo de permanência em um estado M_i, dado que o próximo estado é M_j, é exponencial com parâmetro r_{ij};

c) As probabilidades de estados constantes das marcações são obtidos pela resolução do sistema de equações lineares:

$$S = \begin{cases} xR = 0 \\ \sum_i x_i = 1 \end{cases}$$

No exemplo adotado, a matriz **R** é apresentada na figura 5.12.

	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
M_0	-5	4	0	0	0	0	0	0	1
M_1	0	-14	10	0	0	0	4	0	0
M_2	0	0	-4	4	0	0	0	0	0
M_3	2	0	0	-6	4	0	0	0	0
M_4	0	2	0	0	-12	10	0	0	0
M_5	0	0	2	0	0	-2	0	0	0
M_6	0	0	0	10	0	0	-12	2	0
M_7	10	0	0	0	0	0	0	-10	0
M_8	1	0	0	0	0	0	0	0	-1

Fig. 5.12. Matriz R das Taxas de Transição Instantâneas.

unesp

O vetor solução do sistema **S** é dado por:

$$\mathbf{x} \cong [0{,}094 \;\; 0{,}035 \;\; 0{,}235 \;\; 0{,}176 \;\; 0{,}059 \;\; 0{,}293 \;\; 0{,}012 \;\; 0{,}002 \;\; 0{,}094]^T$$

Definição 5.2: **A matriz $\mathbf{A} = (a_{i,j})$ das probabilidades de próximo estado é dada por:**

$$a_{ij} = - r_{ij} / r_{ij}, \text{ para } i \neq j;$$
$$a_{ij} = 0, \text{ para } i = j.$$

O elemento a_{ij} é a probabilidade de, estando na marcação M_i, alcançar a marcação M_j por meio do disparo de uma transição.

	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
M_0	0	0,8	0	0	0	0	0	0	0,2
M_1	0	0	0,71	0	0	0	0,29	0	0
M_2	0	0	0	1	0	0	0	0	0
M_3	0,33	0	0	0	0,67	0	0	0	0
M_4	0	0,17	0	0	0	0,83	0	0	0
M_5	0	0	1	0	0	0	0	0	0
M_6	0	0	0	0,83	0	0	0	0,17	0
M_7	1	0	0	0	0	0	0	0	0
M_8	1	0	0	0	0	0	0	0	0

Fig. 5.13. Matriz A das Probabilidades de Próximo Estado.

A **função densidade de probabilidade de *token*** num dado estado associa a probabilidade de se ter um certo número de *tokens* em um lugar em uma dada marcação. Tal função pode ser calculada diretamente do vetor **x**. No exemplo adotado, a probabilidade de uma mensagem estar no canal de comunicação é:

$$\mathbf{P}\{1 \textit{ token} \text{ em } \mathbf{MSG}\} = \mathbf{x_1} + \mathbf{x_2} + \mathbf{x_4} + \mathbf{x_5} \cong 0{,}622$$

O **número médio de *tokens* em um conjunto de lugares** é a média ponderada do número de *tokens* num dado conjunto de lugares em marcações distintas com as probabilidades fixas dos estados correspondentes. No exemplo adotado, o número médio de *tokens* nos lugares **2** e **6** é dado por:

$$\mathbf{x_1} + \mathbf{x_3} + 2\mathbf{x_4} + \mathbf{x_5} + 2\mathbf{x_6} + \mathbf{x_7} \cong 0{,}648$$

O **fluxo de *tokens* por uma transição** é dado pela multiplicação da taxa de disparo da transição pela sua probabilidade de um dado estado. Na rede-exemplo, o fluxo através da transição **2** é:

$$\lambda_2 (\mathbf{x_0} + \mathbf{x_3}) \cong 1{,}08 \text{ (}\textit{tokens}\text{ / unidade de tempo)}$$

O **tempo médio para retornar-se a uma marcação $\mathbf{M_i}$** é:

$$\tau_m(\mathbf{M_i}) = (\Sigma_j \lambda_j)^{-1}$$

onde **j** varia sobre o conjunto de transições habilitadas em $\mathbf{M_i}$. No exemplo:

$$\tau_m(\mathbf{M_6}) = (\lambda_1 + \lambda_4)^{-1} \cong 0{,}0833$$

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

CAPÍTULO 6: REDES DE PETRI COLORIDAS

6.1. INTRODUÇÃO INFORMAL

As Redes de Petri Coloridas (RPC) constituem uma das chamadas Redes de Petri de alto-nível. As RP Predicado/Transição (RP Pr/T) foram o primeiro tipo de rede de alto-nível, construídas sem qualquer aplicação particular em mente [12]. RP Pr/T são uma generalização das RP Lugar/Transição, relacionando-se às mesmas de uma maneira formal, tornando possível generalizar a maior parte dos conceitos e métodos de análise que foram desenvolvidos para as tais redes.

Entretanto, logo tornou-se aparente que as RP Pr/T apresentam alguns problemas técnicos, quando os métodos de análise de invariantes de lugar e de transição são generalizados para esse modelo de rede. É possível calcular invariantes para RP Pr/T, mas a interpretação dos mesmos é difícil e deve ser feita com muito cuidado, para evitar resultados errôneos. Na tentativa de superar este problema, Kurt Jensen definiu a primeira versão de Redes de Petri Coloridas, em 1981 (RPC-81) [15]. As idéias principais desse modelo foram diretamente inspiradas nas RP Pr/T, com os arcos da rede sendo rotulados com funções ao invés de expressões, como nas RP Pr/T, significando que os invariantes podiam ser interpretados sem problemas.

No entanto, tornou-se patente que as funções presentes nos arcos das RPC-81 são mais difíceis de ler e entender do que as expressões anexadas aos arcos das RP Pr/T. O forte relacionamento entre os dois modelos de rede levou à idéia da construção de um modelo melhorado, combinando as qualidades de ambas. Este modelo foi definido por K.Jensen [16], sendo chamado Redes de Petri de Alto-Nível (RPAN). Entretanto, este nome deu origem a muita confusão, uma vez que o termo "redes de alto-nível" começou a ser usado na época como um nome genérico para RP Pr/T, RPC-81, RPAN e muitos outros modelos de rede. Para evitar confusão, foi necessário mudar o nome de RPAN para Redes de Petri Coloridas (RPC).

RPC tem duas diferentes representações. A *representação com expressões* usa expressões nos arcos e guardas (predicados) nas transições, enquanto a *representação com funções* usa funções lineares com multi-conjuntos. Há translações formais entre as duas representações. A representação com expressões constitui-se num modelo próximo às RP Pr/T, enquanto a representação com funções aproxima-se às RPC-81.

Atualmente, a maior parte das aplicações práticas de Redes de Petri usam RP Pr/T ou RPC. Há pequenas diferenças entre os dois tipos de rede. As principais diferenças encontram-se escondidas nos métodos utilizados para calcular invariantes de lugar e de transição [19].

Muitas outras classes de Redes de Petri de alto-nível foram definidas. A maior parte delas é similar às RPC, mas utilizando diferentes linguagens de descrição. Alguns dos mais importantes artigos sobre redes de alto-nível, seus métodos de análise e aplicações podem ser encontrados em [17].

A seguir, as RPC serão tratadas em sua representação com funções.

Para introduzir as RPC, será utilizado um exemplo [19] que modela um pequeno sistema de banco de dados (SBD) distribuído (figura 6.1).

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

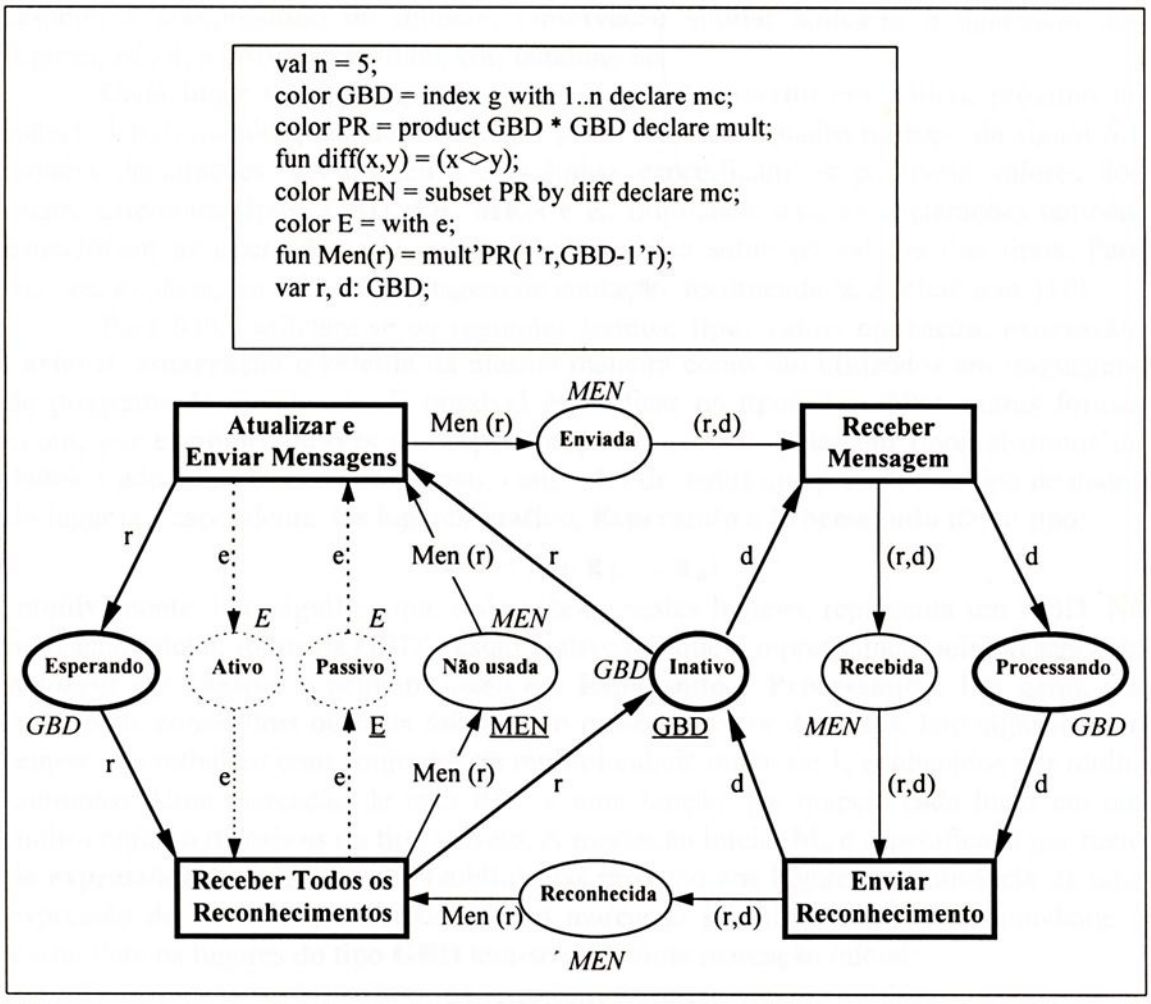


Fig. 6.1. Modelo RPC de um Sistema de Banco de Dados Distribuído.

O SBD tem **n** diferentes lugares geográficos (**n** ≥ 3). Cada lugar contém uma cópia completa da base de dados, a qual é gerenciada por um gerenciador de banco de dados (GBD) local. Quando um GBD $\mathbf{g_i}$ faz uma atualização sobre sua cópia da base de dados, ele deve enviar uma mensagem para todos os outros GBD's, a fim de garantir a consistência das **n** cópias da base de dados. Não levando em conta o conteúdo de cada mensagem, cada qual pode ser representada por um par (**r**,**d**), onde **r** representa o remetente e **d** representa o destinatário. Isto significa que o GBD $\mathbf{g_i}$ envia as seguintes mensagens:

$$\mathbf{Men}(g_i) = \{(g_i,g_1), (g_i,g_2), ..., (g_i,g_{i-1}), (g_i,g_{i+1}), ..., (g_i,g_{n-1}), (g_i,g_n)\}$$

Os estados da RPC são representados por **lugares** (desenhados como elipses). No SBD há nove estados diferentes. Três dos lugares representam três possíveis estados dos GBD's: **Inativo**, **Esperando** e **Processando**. Quatro dos lugares representam os possíveis estados das mensagens: **Não Usada**, **Enviada**, **Recebida** e **Reconhecida**. Os dois lugares remanescentes indicam se uma atualização está sendo feita ou não: **Ativo** e **Passivo**. Por convenção, os nomes são escritos dentro das elipses. Os nomes não têm qualquer significado formal, possuindo, porém, grande importância prática no que diz

respeito à compreensão do modelo. Observação similar aplica-se à aparência dos lugares, isto é, a grossura da linha, cor, tamanho etc.

Cada lugar tem um **tipo** de dados associado (escrito em itálico, próximo ao lugar), determinando quais dados o lugar pode conter. O quadro no topo da figura 6.1 contém declarações. As primeiras seis linhas especificam os possíveis valores dos quatro diferentes tipos: **GBD**, **PR**, **MEN** e **E**. Implicitamente, as declarações também especificam as operações que podem ser efetuadas sobre os valores dos tipos. Para maiores explicações sobre a linguagem de anotação, recomenda-se a referência [18].

Para RPC, utilizam-se os seguintes termos: **tipo**, **valor**, **operação**, **expressão**, **variável**, **amarração** e **cálculo** da mesma maneira como são utilizados em linguagens de programação funcionais. É possível especificar os tipos de muitas outras formas como, por exemplo, através das especificações que são usadas em tipos abstratos de dados. Cada *token* (senha) traz consigo um valor de dados que pertence ao tipo de dados do lugar correspondente. Os lugares **Inativo**, **Esperando** e **Processando** têm o tipo:

$$\mathbf{GBD} = \{\mathbf{g}_1, \mathbf{g}_2, ..., \mathbf{g}_n\}.$$

Intuitivamente, isto significa que cada *token*, nestes lugares, representa um GBD. Na marcação inicial, todos os GBD's estão inativos, o que é representado pela presença de **n** *tokens* em **Inativo** e nenhum *token* em **Esperando** e **Processando**. Em geral, um lugar pode conter dois ou mais *tokens* com o mesmo valor de dados. Isto significa que tem-se que trabalhar com conjuntos de multiplicidade maior de 1, conhecidos por multi-conjuntos. Uma marcação de uma RPC é uma função que mapeia cada lugar em um multi-conjunto de *tokens* do tipo correto. A marcação inicial $\mathbf{M_0}$ é especificada por meio de **expressões de inicialização** (sublinhadas próximo aos lugares). A ausência de uma expressão de inicialização implica que a marcação inicial do lugar correspondente é vazia. Para os lugares do tipo **GBD** tem-se a seguinte marcação inicial:

$$\mathbf{M_0(Inativo)} = \mathbf{GBD}$$
$$\mathbf{M_0(Esperando)} = \mathbf{M_0(Processando)} = \varnothing,$$

onde $\varnothing$ denota o multi-conjunto vazio. Por convenção usa-se **GBD** para denotar o tipo, porém pode ser usado também para denotar o conjunto que contém todos os valores de dados do tipo e o multi-conjunto que contém exatamente uma ocorrência de cada valor de dados do tipo. Conseqüentemente, tem-se:

$$\mathbf{M_0(Inativo)} = 1'\mathbf{g}_1 + 1'\mathbf{g}_2 + ... + 1'\mathbf{g}_n,$$

onde o coeficiente inteiro (antes de ') indica que **Inativo** tem um *token* para cada valor de dados $\mathbf{g}_i$ no tipo **GBD**.

Não Usada, **Enviada**, **Recebida** e **Reconhecida** têm o tipo **MEN**, o qual é um subtipo do produto cartesiano **GBD** × **GBD**. Intuitivamente, isto significa que cada *token* nestes lugares representam uma mensagem (**r**,**d**). Na marcação inicial, todas as mensagens são não usadas. Conseqüentemente, tem-se:

$$\mathbf{M_0(Não\ Usada)} = \mathbf{MEN} = \{(\mathbf{r},\mathbf{d}) \in \mathbf{GBD} \times \mathbf{GBD} \mid \mathbf{r} \neq \mathbf{d}\}$$
$$\mathbf{M_0(Enviada)} = \mathbf{M_0(Recebida)} = \mathbf{M_0(Reconhecida)} = \varnothing,$$

onde a exigência $\mathbf{r} \neq \mathbf{d}$ indica que não se tem mensagens onde o remetente e o destinatário sejam os mesmos.

Ativo e **Passivo** têm o tipo $\mathbf{E} = \{\mathbf{e}\}$. Sua marcação inicial é:

$$\mathbf{M_0(Passivo)} = \mathbf{E} = 1'\mathbf{e}$$
$$\mathbf{M_0(Ativo)} = \varnothing.$$

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Por razões históricas, a teoria de RPC refere-se aos valores de *tokens* como **cores** de *tokens* e aos tipos de dados como **conjuntos de cores**. Os tipos de uma RPC podem ser arbitrariamente complexos como, por exemplo, um registro no qual um campo é um real, outro uma *string* de texto e um terceiro uma lista de inteiros. Assim, é muito mais adequado imaginar um *continuum* de cores ao invés de um conjunto discreto de cores.

As ações de uma RPC são representadas por meio de **transições** (representadas por retângulos). No SBD da figura 6.1, há quatro transições diferentes. Um arco com origem em um lugar e término em uma transição indica que a transição pode remover *tokens* do respectivo lugar, enquanto um arco saindo de uma transição para um lugar indica que a transição pode adicionar *tokens*. O número exato de *tokens* e seus valores de dados são determinados pelas **expressões de arcos** (junto aos arcos). A transição **Atualizar e Enviar Mensagens (EM)** tem seis arcos com três diferentes expressões de arcos: **e**, **r** e **Men(r)**. A primeira delas é constante. As duas outras possuem uma variável livre **r** do tipo **GBD**, declarada na última linha das declarações. Para falar sobre o disparo de uma transição **EM** necessita-se amarrar (associar) **r** a um valor de **GBD**. Caso contrário, não se pode calcular as expressões **r** e **Men(r)**, sendo que esta última mapeia cada GBD **r** às mensagens que **r** envia.

Assumindo que a variável **r** é amarrada ao valor $\mathbf{g}_2$, para a transição **EM**, tem-se a seguinte notação: (**EM**,<**r** = $\mathbf{g}_2$>), a qual é nomeada **elemento de amarração**. Para este elemento de amarração, calculam-se as expressões de arcos como segue (daqui em diante será assumido que **n** = 5 GBD):

$$\mathbf{e} \rightarrow \mathbf{e}$$

$$\mathbf{r} \rightarrow \mathbf{g}_2$$

$$\mathbf{Men(r)} \rightarrow 1'(\mathbf{g}_2,\mathbf{g}_1) + 1'(\mathbf{g}_2,\mathbf{g}_3) + 1'(\mathbf{g}_2,\mathbf{g}_4) + 1'(\mathbf{g}_2,\mathbf{g}_5).$$

Isto significa que o disparo da transição **EM** com a amarração <**r** = $\mathbf{g}_2$>, removerá um *token* com um valor **e** de **Passivo** e adicionará um *token* com valor **e** em **Ativo**. O disparo removerá, também, um *token* com valor $\mathbf{g}_2$ de **Inativo** e adicionará um *token* de igual valor à **Esperando**. Finalmente, removerá quatro *tokens* de **Não Usada** (com os valores especificados pelo cálculo de **Men(r)**) e adicionará quatro *tokens* com estes valores em **Enviada**. Intuitivamente, isto significa que o GBD $\mathbf{g}_2$ muda do estado passivo para ativo e, simultaneamente, envia uma mensagem para cada um dos outros GBD's.

O disparo do elemento de amarração (**EM**,<**r** = $\mathbf{g}_2$>) é possível se os seis *tokens* a serem removidos existem, isto é, se **Passivo** tem um *token* **e**, **Inativo** tem um *token* $\mathbf{g}_2$ e **Não Usada** tem os quatro *tokens* especificados por **Men**($\mathbf{g}_2$). Este é o caso da marcação inicial e, assim, diz-se que o elemento de amarração (**EM**,<**r** = $\mathbf{g}_2$>) está **habilitado** em $\mathbf{M}_0$. É fácil verificar que todas as cinco possíveis amarrações para **EM** geram elementos de amarração habilitados em $\mathbf{M}_0$.

As outras três transições trabalham de forma similar. As duas transições à direita usam duas variáveis diferentes, **r** e **d**, tomadas de **GBD**. Na marcação inicial, todas as possíveis amarrações para as transições **Receber Mensagem (RM)**, **Receber Todos os Reconhecimentos (RT)** e **Enviar Reconhecimento (ER)**, estão desabilitadas. Para **RM**, não há *token* em **Enviada**. Para **ER** e **RT**, não há *token* em qualquer de seus lugares de entrada. Assim, conclui-se que a única coisa que pode acontecer em $\mathbf{M}_0$ é o disparo de um dos cinco elementos de amarração de **EM**. Cada um deles está habilitado e a escolha entre qual deles deva disparar é não-determinística. Tão logo um elemento

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

de amarração dispare, os outros tornam-se desabilitados devido a ausência de um *token* em **Passivo**. Neste caso, diz-se que os elementos de amarração estão em **conflito** entre si.

Suponha que o elemento de amarração (**EM**,<**r** = $\mathbf{g}_2$>) dispare, passando a rede para uma marcação $\mathbf{M}_1$, com os seguintes *tokens* (lista-se apenas os lugares não vazios):

$\mathbf{M}_1$(**Inativo**) = **GBD** -1'$\mathbf{g}_2$ = 1'$\mathbf{g}_1$ + 1'$\mathbf{g}_3$ + 1'$\mathbf{g}_4$ + 1'$\mathbf{g}_5$
$\mathbf{M}_1$(**Esperando**) = 1'$\mathbf{g}_2$
$\mathbf{M}_1$(**Enviada**) = **Men**($\mathbf{g}_2$) = 1'($\mathbf{g}_2$,$\mathbf{g}_1$) + 1'($\mathbf{g}_2$,$\mathbf{g}_3$) + 1'($\mathbf{g}_2$,$\mathbf{g}_4$) + 1'($\mathbf{g}_2$,$\mathbf{g}_5$)
$\mathbf{M}_1$(**Não Usada**) = **MEN** - **Men**($\mathbf{g}_2$)
$\mathbf{M}_1$(**Ativo**) = 1'**e**.

Na marcação $\mathbf{M}_1$ a transição **RM** está habilitada para todas as quatro possíveis amarrações na qual **r** está limitada a $\mathbf{g}_2$ enquanto **d** está restrito aos outros quatro valores de **GBD** distintos de $\mathbf{g}_2$. Isto significa que todos os outros gerenciadores estão prontos para receber as mensagens enviadas por $\mathbf{g}_2$. O elemento de amarração (**RM**,<**r** = $\mathbf{g}_2$, **d** = $\mathbf{g}_3$>) move um *token* com valor ($\mathbf{g}_2$,$\mathbf{g}_3$) de **Enviada** para **Recebida** e move um *token* com valor $\mathbf{g}_3$ de **Inativo** para **Processando**. Isto significa que o GBD $\mathbf{g}_3$ muda do estado inativo para o de processando e, simultaneamente, a mensagem ($\mathbf{g}_2$,$\mathbf{g}_3$) muda do estado enviada para o recebida. Analogamente, o elemento de amarração (**RM**,<**r** = $\mathbf{g}_2$, **d** = $\mathbf{g}_4$>) move um *token* com valor ($\mathbf{g}_2$,$\mathbf{g}_4$) de **Enviada** para **Recebida** e move um *token* com valor $\mathbf{g}_4$ de **Inativo** para **Processando**. Os dois elementos de amarração tratam de *tokens* diferentes e, conseqüentemente, não influem um no outro. Isto significa que eles podem disparar **concorrentemente** um com outro, isto é, no mesmo **passo**. É fácil verificar que todos os quatro elementos de amarração de **RM** são concorrentes entre si. Isto significa que o próximo passo pode conter todos eles ou qualquer subconjunto não-vazio deles. Novamente, a escolha é não-determinística.

As relações de conflito e de concorrência entre elementos de amarração são dependentes de marcação. Assim, uma situação de conflito pode tornar-se numa de concorrência com a simples adição de *tokens*. Por exemplo, poderia-se mudar a marcação inicial de **Passivo** de 1'**e** para 3'**e**, permitindo, portanto, passos onde até três elementos de amarração de **RM** disparem concorrentemente. É também permitido que um elemento de amarração possa ser concorrente com ele próprio, uma ou mais vezes. Isto significa que, em geral, um passo é um multi-conjunto de elementos de amarração.

Suponha que o passo:

1'(**RM**,<**r** = $\mathbf{g}_2$,**d** = $\mathbf{g}_3$>) + 1'(**RM**,<**r** = $\mathbf{g}_2$,**d** = $\mathbf{g}_4$>)

dispare na marcação $\mathbf{M}_1$, gerando a marcação $\mathbf{M}_2$:

$\mathbf{M}_2$(**Inativo**) = $\mathbf{M}_1$(**Inativo**) - (1'$\mathbf{g}_3$ + 1'$\mathbf{g}_4$) = 1'$\mathbf{g}_1$ + 1'$\mathbf{g}_5$
$\mathbf{M}_2$(**Esperando**) = 1'$\mathbf{g}_2$
$\mathbf{M}_2$(**Processando**) = 1'$\mathbf{g}_3$ + 1'$\mathbf{g}_4$
$\mathbf{M}_2$(**Enviada**) = $\mathbf{M}_1$(**Enviada**) - (1'($\mathbf{g}_2$,$\mathbf{g}_3$) + 1'($\mathbf{g}_2$,$\mathbf{g}_4$)) = 1'($\mathbf{g}_2$,$\mathbf{g}_1$) + 1'($\mathbf{g}_2$,$\mathbf{g}_5$)
$\mathbf{M}_2$(**Recebida**) = 1'($\mathbf{g}_2$,$\mathbf{g}_3$) + 1'($\mathbf{g}_2$,$\mathbf{g}_4$)
$\mathbf{M}_2$(**Não Usada**) = **MEN** - **Men**($\mathbf{g}_2$)
$\mathbf{M}_2$(**Ativo**) = 1'**e**.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Em $\mathbf{M_2}$, há quatro elementos de amarração habilitados:

(RM,<r = g_2,d = g_1>)
(RM,<r = g_2,d = g_5>)
(ER,<r = g_2,d = g_3>)
(ER,<r = g_2,d = g_4>).

Os quatro elementos de amarração são concorrentes entre si, uma vez que utilizam *tokens* diferentes. Suponha que os três primeiros disparem, obtendo-se, assim, o passo:

$$1'(\mathbf{RM},<\mathbf{r} = g_2,\mathbf{d} = g_1>) + 1'(\mathbf{RM},<\mathbf{r} = g_2,\mathbf{d} = g_5>) + 1'(\mathbf{ER},<\mathbf{r} = g_2,\mathbf{d} = g_3>)$$

A RPC passaria para uma marcação $\mathbf{M_3}$:

$\mathbf{M_3(Inativo)} = (\mathbf{M_2(Inativo)} - (1'g_1 + 1'g_5)) + 1'g_3 = 1'g_3$
$\mathbf{M_3(Esperando)} = 1'g_2$
$\mathbf{M_3(Processando)} = (\mathbf{M_2(Processando)} - 1'g_3) + (1'g_1 + 1'g_5)$
$= 1'g_1 + 1'g_4 + 1'g_5$
$\mathbf{M_3(Enviada)} = \mathbf{M_2(Enviada)} - (1'(g_2,g_1) + 1'(g_2,g_5)) = \varnothing$
$\mathbf{M_3(Recebida)} = (\mathbf{M_2(Recebida)} - 1'(g_2,g_3)) + (1'(g_2,g_1) + 1'(g_2,g_5))$
$= 1'(g_2,g_1) + 1'(g_2,g_4) + 1'(g_2,g_5)$
$\mathbf{M_3(Reconhecida)} = 1'(g_2,g_3)$
$\mathbf{M_3(Não\ Usada)} = \mathbf{MEN} - \mathbf{Men}(g_2)$
$\mathbf{M_3(Ativo)} = 1'\mathbf{e}.$

Em $\mathbf{M_3}$, há três elementos de amarração habilitados:

(ER,<r = g_2,d = g_1>)
(ER,<r = g_2,d = g_4>)
(ER,<r = g_2,d = g_5>).

O disparo concorrente desses três elementos de amarração leva a uma marcação $\mathbf{M_4}$:

$\mathbf{M_4(Inativo)} = 1'g_1 + 1'g_3 + 1'g_4 + 1'g_5$
$\mathbf{M_4(Esperando)} = 1'g_2$
$\mathbf{M_4(Reconhecida)} = 1'(g_2,g_1) + 1'(g_2,g_3) + 1'(g_2,g_4) + 1'(g_2,g_5) = \mathbf{Men}(g_2)$
$\mathbf{M_4(Não\ Usada)} = \mathbf{MEN} - \mathbf{Men}(g_2)$
$\mathbf{M_4(Ativo)} = 1'\mathbf{e}.$

Em $\mathbf{M_4}$, há apenas um elemento de amarração habilitado:

(RT,<r = g_2>).

O disparo deste elemento de amarração leva à marcação inicial $\mathbf{M_0}$. As marcações e passos considerados anteriormente formam uma **seqüência de disparos**:

$$\mathbf{M_0}\ [\mathbf{P_0}>\ \mathbf{M_1}\ [\mathbf{P_1}>\ \mathbf{M_2}\ [\mathbf{P_2}>\ \mathbf{M_3}\ [\mathbf{P_3}>\ \mathbf{M_4}\ [\mathbf{P_4}>\ \mathbf{M_0}.$$

A seqüência de disparos adotada é apenas uma dentre muitas outras.

Para garantir a consistência entre as diferentes cópias da base de dados, o modelo simplesmente permite apenas uma atualização por vez. Quando um GBD inicializa uma atualização, a mesma tem que ser realizada por todos os outros GBD's antes que qualquer outra atualização possa ser iniciada. A exclusão mútua é garantida pelo lugar **Passivo**. A descrição do SBD é muito alto-nível (e fora da realidade), no

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

sentido que a exclusão mútua é descrita através de um mecanismo global (o lugar **Passivo**). Para implementar o SBD em *hardware* distribuído, a exclusão mútua pode ser tratada através de um "mecanismo distribuído". Tal implementação poderia ser descrita por uma RPC mais detalhada.

A RPC para o SBD tem vários lugares redundantes, os quais poderiam ser omitidos sem mudar o comportamento do sistema. Como exemplo, poderia-se omitir o lugar **Não Usada**. Assim, haveria somente a representação explícita daquelas mensagens que estão correntemente em uso (lugares **Enviada**, **Recebida** e **Reconhecida**). Da mesma forma, poderia-se omitir os lugares **Ativo** e **Processando**. A redundância de lugares pode ser justificada pela melhor clareza do modelo, ficando, portanto, a cargo do modelador do sistema.

A atribuição de valores de dados a cada *token* permite o uso de muito menos lugares do que aqueles necessários por uma RP de baixo-nível. Para o SBD, uma RP clássica teria **n** lugares para cada um dos estados **Inativo**, **Esperando** e **Processando**, uma vez que esta seria a única maneira da RP clássica distinguir entre os **n** diferentes GBD's. Analogamente, seriam necessários $|\mathbf{MEN}| = \mathbf{n}^2 - \mathbf{n}$ lugares diferentes para os estados referentes às mensagens (**Não Usada**, **Enviada**, **Recebida** e **Reconhecida**). Desta forma, uma RP clássica necessitaria de 97 lugares para representar um SBD com 5 GBD's, enquanto uma RPC necessita apenas de 9 lugares.

O uso de variáveis nas expressões dos arcos significa que cada transição de uma RPC pode disparar de vária maneiras. Assim, pode-se utilizar uma única transição para descrever uma classe de atividades relacionadas, enquanto numa RP clássica seria necessário uma transição para cada uma dessas atividades. No SBD, há somente uma transição **EM** e somente uma transição **RT**. Na RP clássica haveria uma transição de cada uma dessas para cada GBD. Analogamente, haveria uma transição **RM** e uma transição **ER** para cada elemento de **MEN**. Assim, com 5 GBD's, a RP clássica teria 50 transições, enquanto a RPC tem apenas 4.

Em adição às expressões de arcos, é possível anexar uma expressão booleana (baseada nas variáveis) a cada transição. Tal expressão é chamada **guarda**. Esta especifica que deve-se aceitar elementos de amarração para os quais a expressão booleana seja verdadeira. No SBD, poderia-se adicionar a guarda:

$$[\mathbf{r} \neq \mathbf{g_3}]$$

à transição **EM**. Tal guarda significaria a exclusão de $(\mathbf{EM}, \langle \mathbf{r} = \mathbf{g}_2 \rangle)$ do conjunto de elementos de amarração e, conseqüentemente, não seria possível o GBD $\mathbf{g_3}$ iniciar uma atualização.

Após esta apresentação informal, prossegue-se a definição formal de RPC não hierárquicas.

6.2. Definição Formal

RPC são matematicamente definidas por uma tupla. Em princípio, é fácil transladar um diagrama RPC em uma tupla RPC e vice-versa.

Inicialmente, será apresentada a definição de multi-conjuntos, onde $\mathbf{N}$ denota o conjunto dos inteiros não-negativos.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Definição 6.1: **Multi-Conjunto.**

Um **multi-conjunto m**, sobre um conjunto não-nulo **S**, é uma função $\mathbf{m}:\mathbf{S}\rightarrow\mathbb{N}$ representando a soma formal:

$$\sum_{s\in S} \mathbf{m(s)}'\mathbf{s}.$$

Denota-se $\mathbf{S_{MC}}$ o conjunto de todos os multi-conjuntos sobre **S**. Os inteiros não-negativos $\{\mathbf{m(s)}: \mathbf{s}\in\mathbf{S}\}$ são os coeficientes do multi-conjunto. $\mathbf{s}\in\mathbf{m}$ se e somente se $\mathbf{m(s)} \neq 0$.

Há uma correspondência 1-1 entre conjuntos sobre **S** e os multi-conjuntos para os quais todos os coeficientes são zero ou um. Será utilizado $\mathbf{A} \subseteq \mathbf{S}$ para denotar o multi-conjunto que consiste de uma única ocorrência para cada elemento em **A**. Analogamente, será usado um elemento $\mathbf{s}\in\mathbf{S}$ para denotar o multi-conjunto 1's.

Existe um conjunto de operações padrão para multi-conjuntos (com exceção de $|\mathbf{m}|$, todas as demais são derivadas das operações padrão para funções).

Definição 6.2: **Operações sobre Multi-Conjuntos.**

Comparação, **adição**, **multiplicação escalar**, e **tamanho** de multi-conjuntos são definidos como segue, para todo $\mathbf{m},\mathbf{m_1},\mathbf{m_2}\in\mathbf{S_{MC}}$ e todo $\mathbf{n}\in\mathbb{N}$:

i) $\mathbf{m_1} \neq \mathbf{m_2} = \exists\ \mathbf{s}\in\mathbf{S}: \mathbf{m_1(s)} \neq \mathbf{m_2(s)}$ (comparação, onde *R* pode ser $\leq$, $\geq$ ou $=$);
$\mathbf{m_1}\ R\ \mathbf{m_2} = \forall\ \mathbf{s}\in\mathbf{S}: \mathbf{m_1(s)}\ R\ \mathbf{m_2(s)}$

ii) $\mathbf{m_1} + \mathbf{m_2} = \sum_{s\in S} (\mathbf{m_1(s)} + \mathbf{m_2(s)})'\mathbf{s}$ (adição)

$\mathbf{m_1} - \mathbf{m_2} = \sum_{s\in S} (\mathbf{m_1(s)} - \mathbf{m_2(s)})'\mathbf{s}$ (subtração: $\mathbf{m_1} \geq \mathbf{m_2}$);

iii) $\mathbf{n} * \mathbf{m} = \sum_{s\in S} (\mathbf{n} * \mathbf{m(s)})'\mathbf{s}$ (multiplicação escalar);

iv) $|\mathbf{m}| = \sum_{s\in S} \mathbf{m(s)}$ (tamanho).

Obs.: Quando $|\mathbf{m}| = \infty$ diz-se que **m** é infinito. Caso contrário, **m** é finito.

As operações com multi-conjuntos têm um grande número de propriedades algébricas padrão. Mais detalhes podem ser encontrados em [18].

Para dar uma definição abstrata de RPC não é necessário fixar a sintaxe na qual o modelador especifica as expressões da rede. A referência [19] assume que tal sintaxe existe juntamente com uma semântica bem definida, tornando possível falar, de uma forma não ambígua, sobre:

a) Os elementos de um tipo **T**. O conjunto de todos os elementos em **T** é denotado pelo próprio nome **T**;
b) O tipo de uma variável **v** - denotado por **Tipo(v)**;
c) O tipo de uma expressão **expr** - denotado por **Tipo(expr)**;
d) O conjunto de variáveis em uma expressão **expr** - denotado por **Var(expr)**;
e) Uma amarração de um conjunto de variáveis **V** - associa-se a cada variável $\mathbf{v}\in\mathbf{V}$ um elemento $\mathbf{b(v)}\in\mathbf{Tipo(v)}$;

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

f) O valor obtido pelo cálculo de uma expressão **expr** em uma amarração **b** - denotado por **expr<b>**. Exige-se que **Var(expr)** seja um subconjunto de variáveis de **b**, sendo o cálculo executado através da substituição de cada variável $\mathbf{v} \in \mathbf{Var(expr)}$ pelo valor $\mathbf{b(v)} \in \mathbf{Tipo(v)}$, determinado pela amarração.

Uma expressão sem variáveis é dita uma **expressão fechada**. Ela pode ser calculada para todas as amarrações e cada cálculo fornece sempre o mesmo valor. Neste caso, o valor obtido pelo cálculo de **expr** em uma amarração **b** será denotado apenas por **expr**.

Será utilizado **B** para denotar o tipo booleano ($\mathbf{B} = \{\mathbf{falso}, \mathbf{verdadeiro}\}$), deste ponto em diante. Além disso, a notação **Tipo(v)** será estendida para $\mathbf{Tipo(A)} = \{\mathbf{Tipo(v)} \mid \mathbf{v} \in \mathbf{A}\}$, onde **A** é um conjunto de variáveis.

Segue-se a definição de RPC. Algumas motivações e explanações de partes individuais da definição são dadas imediatamente após a definição.

Definição 6.3: **Representação Matemática de RPC.**

Uma **RPC** é uma tupla $RPC = (\Sigma, \mathbf{L}, \mathbf{T}, \mathbf{A}, \mathbf{N}, \mathbf{C}, \mathbf{G}, \mathbf{E}, \mathbf{I})$, onde:

i) Σ é um conjunto finito não-nulo de **tipos**, também chamados conjuntos de cores;

ii) **L** é um conjunto finito de **lugares**;

iii) **T** é um conjunto finito de **transições**;

iv) **A** é um conjunto finito de **arcos**, de forma que:

$$\mathbf{L} \cap \mathbf{T} = \mathbf{L} \cap \mathbf{A} = \mathbf{T} \cap \mathbf{A} = \varnothing$$

v) **N** é uma função **nó**, definida de **A** em $\mathbf{L} \times \mathbf{T} \cup \mathbf{T} \times \mathbf{L}$;

vi) **C** é uma função **cor**, definida de **L** em Σ;

vii) **G** é uma função **guarda**, definida de **T** em expressões, de forma que:

$$\forall\, \mathbf{t} \in \mathbf{T}: [\mathbf{Tipo(G(t))} = \mathbf{B} \wedge \mathbf{Tipo(Var(G(t)))} \subseteq \Sigma];$$

viii) **E** é uma função de **expressão de arco**, definida de **A** em expressões, de forma que:

$$\forall\, \mathbf{a} \in \mathbf{A}: [\mathbf{Tipo(E(a))} = \mathbf{C(l)_{MC}} \wedge \mathbf{Tipo(Var(E(a)))} \subseteq \Sigma]$$

onde **l** é o lugar de **N(a)**;

ix) **I** é uma função de **inicialização**, definida de **L** em expressões fechadas, de forma que:

$$\forall\, \mathbf{l} \in \mathbf{L}: [\mathbf{Tipo(I(l))} = \mathbf{C(l)_{MC}}].$$

Observações:

i) O conjunto de **tipos** determina os valores de dados e as operações e funções que podem ser usadas nas expressões da rede, isto é, expressões de arcos, guardas e expressões de inicialização. Assume-se que cada tipo possui pelo menos um elemento;

ii) + iii) + iv) Os **lugares**, **transições** e **arcos** são descritos pelos conjuntos **L**, **T** e **A**, finitos e disjuntos dois a dois. A exigência destes conjuntos serem finitos evita uma grande número de problemas técnicos, tais como a possibilidade de existir um número infinito de arcos entre dois nós;

unesp

v) A função **nó** mapeia cada arco em um par onde o primeiro elemento é o nó origem e o segundo elemento é o nó destino. Os dois nós devem ser de tipos diferentes, isto é, um deve ser um lugar e o outro uma transição. RPC permitem a existência de vários arcos entre o mesmo par ordenado de nós;

vi) A função **cor C** mapeia cada lugar **l** a um tipo **C(l)**. Intuitivamente, isto significa que cada *token* em **l** deve ter um valor de dados que pertença a **C(l)**;

vii) A função **guarda G** mapeia cada transição **t** em uma expressão booleana onde todas as variáveis têm tipos que pertençam a Σ. Expressões boolenas sempre verdadeiras são omitidas;

viii) A função de **expressão de arco E** mapeia cada arco **a** em uma expressão do tipo $\mathbf{C(l)_{MC}}$. Isto significa que cada expressão de arco deve ser calculada sobre os multi-conjuntos do tipo do lugar adjacente **l**;

ix) A função de **inicialização I** mapeia cada lugar **l** em uma expressão fechada que deve ser do tipo $\mathbf{C(l)_{MC}}$. Funções de inicialização que resultam em vazio são omitidas.

Após a definição da estrutura das RPC, passa-se às considerações sobre seu comportamento. Para tal, inicialmente, será introduzida a seguinte notação para todo $\mathbf{t}\in\mathbf{T}$ e todo par de nós $(\mathbf{x_1},\mathbf{x_2}) \in (\mathbf{L}\times\mathbf{T} \cup \mathbf{T}\times\mathbf{L})$:

a) $\mathbf{A(t)} = \{\mathbf{a}\in\mathbf{A} \mid \mathbf{N(a)} \in \mathbf{L}\times\{\mathbf{t}\} \cup \{\mathbf{t}\}\times\mathbf{L}\}$;

b) $\mathbf{Var(t)} = \{\mathbf{v} \mid \mathbf{v} \in \mathbf{Var(G(t))} \vee \exists\, \mathbf{a}\in\mathbf{A(t)}: \mathbf{v} \in \mathbf{Var(E(a))}\}$;

c) $\mathbf{A(x_1,x_2)} = \{\mathbf{a}\in\mathbf{A} \mid \mathbf{N(a)} = (\mathbf{x_1},\mathbf{x_2})\}$;

d) $\mathbf{E(x_1,x_2)} = \sum_{\mathbf{a}\in\mathbf{A(x_1,x_2)}} \mathbf{E(a)}$.

A soma em d) indica adição de expressões. A partir dos argumentos, está sempre claro se trata-se da função $\mathbf{E}\in[\mathbf{A}\rightarrow\mathbf{Expr}]$ ou da função $\mathbf{E}\in[(\mathbf{L}\times\mathbf{T} \cup \mathbf{T}\times\mathbf{L})\rightarrow\mathbf{Expr}]$. Uma observação similar aplica-se à **A**, **A(t)** e $\mathbf{A(x_1,x_2)}$.

A seguir, definem-se amarrações. Deve-se notar que ii) implica que todas as amarrações satisfazem a guarda correspondente (**G(t)<b>** denota o cálculo da expressão guarda **G(t)** na amarração **b**).

Definição 6.4: **Amarração.**
Uma **amarração** de uma transição **t** é uma função **b** definida sobre **Var(t)**, tal que:
i) $\forall\mathbf{v}\in\mathbf{Var(t)}: \mathbf{b(v)}\in \mathbf{Tipo(v)}$;
ii) **G(t)<b>**.
Denota-se **B(t)** o conjunto de todas as amarrações para **t**.

Conforme mostrado na seção 6.1, amarrações são escritas geralmente na forma $\langle \mathbf{v_1} = \mathbf{c_1}, \mathbf{v_2} = \mathbf{c_2},..., \mathbf{v_n} = \mathbf{c_n}\rangle$, onde $\mathbf{Var(t)} = \{\mathbf{v_1}, \mathbf{v_2}, ..., \mathbf{v_n}\}$. A ordem das variáveis não tem importância.

Definição 6.5: Um **elemento de *token*** é um par **(l,c)** onde $\mathbf{l}\in\mathbf{L}$ e $\mathbf{c}\in\mathbf{C(l)}$, enquanto um **elemento de amarração** é um par **(t,b)** onde $\mathbf{t}\in\mathbf{T}$ e $\mathbf{b}\in\mathbf{B(t)}$. O

unesp

conjunto de todos os elementos de *token* é denotado por **ET** enquanto o conjunto de todos os elementos de amarração é denotado por **EA**.
Uma **marcação** é um multi-conjunto sobre **ET** enquanto um **passo** é um multi-conjunto não-nulo e finito sobre **EA**. A marcação inicial $\mathbf{M_0}$ é a marcação obtida do cálculo das expressões de inicilização:

$$\forall(\mathbf{l},\mathbf{c})\in\mathbf{ET}: \mathbf{M_0}(\mathbf{l},\mathbf{c}) = (\mathbf{I}(\mathbf{l}))(\mathbf{c}).$$

Os conjuntos de todas as marcações e passos são denotados por **M** e **P**, respectivamente.

Cada marcação $\mathbf{M}\in\mathbf{ET_{MC}}$ determina uma única função **M*** definida sobre **L**, tal que $\mathbf{M^*(l)}\in\mathbf{C(l)_{MC}}$:

$$\forall\ \mathbf{l}\in\mathbf{L},\ \forall\ \mathbf{c}\in\mathbf{C(l)}: (\mathbf{M^*(l)})(\mathbf{c}) = \mathbf{M(l,c)}.$$

Por outro lado, cada função **M*** definida sobre **L**, tal que $\mathbf{M^*(l)}\in\mathbf{C(l)_{MC}}$ para todo $\mathbf{l}\in\mathbf{L}$, determina uma única marcação **M**:

$$\forall\ (\mathbf{l},\mathbf{c})\in\mathbf{ET}: \mathbf{M(l,c)} = (\mathbf{M^*(l)})(\mathbf{c}).$$

Desta forma, marcações são geralmente representadas por funções definidas sobre **L**.

A seguir, é dada a definição formal de habilitação e disparo.

Definição 6.6: Um passo **P** está habilitado em uma marcação **M** se e somente se a seguinte propriedade for satisfeita:

$$\forall\ \mathbf{l}\in\mathbf{L}: \sum_{(\mathbf{t},\mathbf{b})\in\mathbf{P}} \mathbf{E(l,t)<b>} \le \mathbf{M(l)}.$$

Então, diz-se que **(t,b)** está habilitado e que **t** está habilitada. Se **P** tiver mais de um elemento, diz-se que os mesmos estão habilitados concorrentemente.
Quando um passo **P** está habilitado em uma marcação $\mathbf{M_1}$ ele pode disparar, mudando a marcação da rede de $\mathbf{M_1}$ para $\mathbf{M_2}$, a qual é definida por:

$$\forall\ \mathbf{l}\in\mathbf{L}: \mathbf{M_2(l)} = (\mathbf{M_1(l)} - \sum_{(\mathbf{t},\mathbf{b})\in\mathbf{P}} \mathbf{E(l,t)<b>}) + \sum_{(\mathbf{t},\mathbf{b})\in\mathbf{P}} \mathbf{E(t,l)<b>}.$$

Diz-se que $\mathbf{M_2}$ é diretamente alcançável de $\mathbf{M_1}$, o que é denotado por:

$$\mathbf{M_1}\ [\mathbf{P}\rangle\ \mathbf{M_2}.$$

O cálculo da expressão **E(l,t)<b>** fornece os *tokens* que são removidos de **l** quando **t** dispara com a amarração **b**. Somando-se todos os elementos de amarração $(\mathbf{t},\mathbf{b})\in\mathbf{P}$, obtém-se todos os *tokens* que são removidos de **l** quando **P** dispara. Exige-se que este multi-conjunto seja menor ou igual à marcação de **l**. Isto significa que cada elemento de amarração $(\mathbf{t},\mathbf{b})\in\mathbf{P}$ deve ser capaz de obter os *tokens* especificados por **E(l,t)<b>**, sem ter que compartilhar estes *tokens* como outros elementos de amarração de **P**.

O disparo de um passo é um evento indivisível. Embora a fórmula anterior exija que a subtração seja realizada antes da adição, não deve ser reconhecida a existência de uma marcação intermediária, onde os *tokens* da primeira soma teriam sido removidos, enquanto aqueles da segunda soma não tivessem sido ainda adicionados. Deve ser observado, também, que um passo não necessita ser maximal. Quando um certo número

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

de elementos de amarração estão habilitados concorrentemente, é possível o disparo de um passo que contém apenas alguns deles.

Definição 6.7: Uma **seqüência de disparos finita** é uma seqüência de marcações e passos:

$$\mathbf{M_1\ [P_1>\ M_2\ [P_2>\ M_3 \ldots\ M_n\ [P_n>\ M_{n+1}}$$

onde $\mathbf{n \in N}$ e $\mathbf{M_1\ [P_i>\ M_{i+1}}$ para todo $\mathbf{i \in \{1,2,\ldots,n\}}$, sendo $\mathbf{M_i}$ a **marcação inicial**, $\mathbf{M_{n+i}}$ a **marcação final** e **n** o **comprimento**.

Analogamente, uma **seqüência de disparos infinita** é uma seqüência de marcações e passos:

$$\mathbf{M_1\ [P_1>\ M_2\ [P_2>\ M_3 \ldots}$$

onde $\mathbf{M_1\ [P_i>\ M_{i+1}}$ para todo $\mathbf{i \geq 1}$.

Uma marcação **M"** é **alcançável de** uma marcação **M'** se e somente se existe uma **seqüência de disparos finita** começando em **M'** e terminando em **M"**. O conjunto de marcações que são alcançáveis de M' é denotado por **[M'>**. Uma marcação é **alcançável** se e somente se pertence a $\mathbf{[M_0>}$.

Seqüências de disparo de tamanho zero são permitidas. Isto significa que $\mathbf{M \in [M>}$ para toda marcação **M**.

6.3. Propriedades Dinâmicas

As propriedades dinâmicas caracterizam o comportamento de uma dada RPC como, por exemplo, se é possível alcançar uma marcação na qual nenhum passo esteja habilitado.

A seguir serão introduzidas algumas das mais importantes propriedades dinâmicas.

As propriedades de **limitabilidade** dizem qual a quantidade máxima de *tokens* que um determinado lugar pode ter:

Definição 6.8: Dados um lugar $\mathbf{l \in L}$, um inteiro não-negativo **n** e um multi-conjunto $\mathbf{m \in C(l)_{MC}}$:

i) **n** é limite inteiro para **l** se e somente se:

$$\mathbf{\forall\ M \in [M_0>:\ |M(l)| \leq n;}$$

ii) **m** é um limite multi-conjunto para **l** se e somente se:

$$\mathbf{\forall\ M \in [M_0>:\ M(l) \leq m.}$$

Para o SBD (figura 6.1) com **n** GBD's, os limites dos lugares encontram-se na figura 6.2.

Observe que os limites inteiro e multi-conjunto são suplementares entre si. De um deles é geralmente possível deduzir informação que não pode ser deduzida de outro e vice-versa. Este é, por exemplo, o caso dos lugares **Inativo** e **Esperando**. Para

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Esperando, o limite inteiro fornece uma informação muito mais precisa do que o limite multi-conjunto. No caso de **Inativo** ocorre o contrário.

Lugar	Limite Multi-Conjunto	Limite Inteiro
Inativo	GBD	n
Esperando	GBD	1
Processando	GBD	n - 1
Não Usada	MEN	n^2 - n
Enviada	MEN	n - 1
Recebida	MEN	n - 1
Reconhecida	MEN	n - 1
Passivo	E	1
Ativo	E	1

Fig. 6.2. Limites Inteiro e Multi-Conjunto do Sistema de Banco de Dados.

As propriedades **caseiras** dizem para quais marcações (ou conjuntos de marcações) é sempre possível de se retornar.

Definição 6.9: **Marcação e Espaço Caseiros.**
Dados uma marcação $\mathbf{M} \in \mathbf{M}$ e um conjunto de marcações $\mathbf{X} \subseteq \mathbf{M}$:
i) **M** é uma **marcação caseira** se e somente se:

$$\forall\, \mathbf{M'} \in [\mathbf{M_0}\rangle: \mathbf{M} \in [\mathbf{M'}\rangle;$$

ii) **X** é um **espaço caseiro** se e somente se:

$$\forall\, \mathbf{M'} \in [\mathbf{M_0}\rangle: \mathbf{X} \cap [\mathbf{M'}\rangle \neq \varnothing.$$

É fácil ver que **M** é uma marcação caseira se e somente se {**M**} é um espaço caseiro. Note que a presença de uma marcação caseira afirma que é possível alcançar um espaço caseiro, entretanto, não garante que se alcance, uma vez que possam existir infinitas seqüências de disparos que não contenham tal espaço caseiro. Uma observação similar aplica-se a espaços caseiros.

Para o SBD é possível mostrar que $\mathbf{M_0}$ é uma marcação caseira. Disto, conclui-se que toda marcação alcançável (decorrente) a partir de $\mathbf{M_0}$ é também uma marcação caseira e, portanto, o conjunto das possíveis marcações da rede é um espaço caseiro.

As propriedades de **vivacidade** dizem se um conjunto de elementos de amarração **X** permanece ativo. Se **X** permanecer ativo, significa que para toda marcação alcançável **M'** a partir de $\mathbf{M_0}$, existe uma seqüência de disparos iniciando em **M'** que contenha um elemento de **X**.

Definição 6.10: Dados uma marcação $\mathbf{M} \in \mathbf{M}$ e um conjunto de elementos de amarração $\mathbf{X} \subseteq \mathbf{EA}$:
i) **M** é **morta** se e somente se nenhum elemento de amarração estiver habilitado, isto é, se e somente se:

$$\forall\, \mathbf{x} \in \mathbf{EA}: \neg \mathbf{M}[\mathbf{x}\rangle;$$

ii) **X** é **morto** em **M** se e somente se nenhum elemento de **X** pode tornar-se habilitado, isto é, se e somente se:

$$\forall\, \mathbf{M'} \in [\mathbf{M}\rangle, \forall\, \mathbf{x} \in \mathbf{X}: \neg \mathbf{M}[\mathbf{x}\rangle;$$

unesp

iii)**X** é **vivo** se e somente se não existe marcação alcançável na qual **X** é morto, isto é, se e somente se:

$$\forall\, \mathbf{M'} \in [\mathbf{M_0}\rangle, \exists\, \mathbf{M''} \in [\mathbf{M'}\rangle \text{ e } \exists\, \mathbf{x} \in \mathbf{X}: \mathbf{M''}[\mathbf{x}\rangle.$$

A propriedade vivacidade apenas exige que os elementos de **X** possam tornar-se habilitados. Assim, podem existir infinitas seqüências de disparos começando em **M'** e que não contenham elementos de **X**. Deve-se observar que **vivo** não é negação de **morto**. Cada conjunto vivo de elementos de amarração é não-morto, mas o oposto não é verdade. Para a transição $\mathbf{t} \in \mathbf{T}$, usa-se $\mathbf{EA(t)} \subseteq \mathbf{EA}$ para denotar o conjunto de todos os elementos de amarração que contenham **t**. Diz-se que **t** é viva ou morta se e somente se **EA(t)** possui a respectiva propriedade. Diz-se, também, que **t** é **estritamente viva** se e somente se $\{\mathbf{x}\}$ é vivo para todo $\mathbf{x} \in \mathbf{EA(t)}$.

As quatro transições do SBD são estritamente vivas. Isto pode ser verificado provando que a marcação inicial é uma marcação caseira e que existe uma seqüência de disparos iniciada em $\mathbf{M_0}$ e que contém todos os elementos de amarração de **EA**.

As propriedades de **eqüitatividade** dizem como geralmente disparam os diferentes elementos de amarração. Para um conjunto de elementos de amarração $\mathbf{X} \subseteq \mathbf{EA}$ e uma seqüência infinita de disparos σ da forma:

$$\sigma = \mathbf{M_1}\, [\mathbf{P_1}\rangle\, \mathbf{M_2}\, [\mathbf{P_2}\rangle\, \mathbf{M_3} \ldots$$

Usa-se $\mathbf{H_{X,i}}(\sigma)$ para denotar o número de elementos de **X** que estão habilitados na marcação $\mathbf{M_i}$. Analogamente, usa-se $\mathbf{D_{X,i}}(\sigma)$ para denotar o numero de elementos de **X** que disparam no passo $\mathbf{P_i}$ (quando um elemento dispara concorrentemente com ele mesmo isto é refletido na contagem).

Usa-se $\mathbf{H_X}(\sigma)$ e $\mathbf{D_X}(\sigma)$ para denotar, respectivamente, o número total de habilitações e disparos em σ, ou seja:

$$\mathbf{H_X}(\sigma) = \sum_{i=1}^{\infty} \mathbf{H_{X,i}}(\sigma) \qquad \mathbf{D_X}(\sigma) = \sum_{i=1}^{\infty} \mathbf{D_{X,i}}(\sigma)$$

Como todos os elementos nos dois somatórios são inteiros não-negativos, é fácil ver que os somatórios devem ser convergentes - ou para um elemento de $\mathbb{N}$ ou para ∞.

Definição 6.11: Seja $\mathbf{X} \subseteq \mathbf{EA}$ um conjunto de elementos de amarração e σ uma seqüência de disparos infinita:

i) **X** é **imparcial** de σ se e somente se ele tiver infinitos disparos, isto é, se e somente se:

$$\mathbf{D_X}(\sigma) = \infty;$$

ii) **X** é **eqüitativo** à σ se e somente se um número infinito de habilitações implica um número infinito de disparos, isto é, se e somente se:

$$\mathbf{H_X}(\sigma) = \infty \Rightarrow \mathbf{D_X}(\sigma) = \infty;$$

iii)**X** é **ajustado** por σ se e somente se uma habilitação persistente implica um disparo, isto é, se e somente se:

$$\forall\, \mathbf{i} \geq 1: [\mathbf{H_{X,i}}(\sigma) \neq 0 \Rightarrow \exists\, \mathbf{k} \geq \mathbf{i}: [\mathbf{H_{X,k}}(\sigma) = 0 \vee \mathbf{D_{X,k}}(\sigma) \neq 0]].$$

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Diz-se que uma transição $t \in T$ é **imparcial**, **eqüitativa** ou **ajustada** se e somente se EA(t) possuir a respectiva propriedade. Diz-se que t é **estritamente imparcial** (**eqüitativa/ajustada**) se e somente se $\{x\}$ é imparcial (eqüitativo/ajustado) para todo $x \in EA(t)$.

Todas as transições do SBD da figura 6.1 são imparciais. **EM** é estritamente ajustada, enquanto as outras três são estritamente eqüitativas.

6.4. Grafo de Disparos

Esta seção trata de **grafos de disparos**, também conhecidos por **grafos de alcançabilidade**, **estado de espaços** ou **máquina de senhas**. A idéia básica por trás de grafos de disparos é construir um grafo com um nó para cada marcação alcançável e um arco para cada elemento de amarração. Obviamente, tal grafo pode tornar-se extremamente grande, mesmo para pequenas redes.

A figura 6.3 mostra um grafo de disparos para o SBD com 3 GBD's. Para economizar espaço, as marcações são representadas pela listagem daqueles GBD's que têm uma mensagem endereçadas para eles - em **Enviada**, **Recebida** ou **Reconhecida**, respectivamente. Isto significa que (2,3,-) denota uma marcação na qual g_1 está **Esperando**, g_2 está **Inativo** e g_3 está **Processando**. Analogamente, (23,-,-) denota uma marcação na qual g_1 está **Esperando**, enquanto g_2 e g_3 estão em **Inativo**. A marcação inicial é representada por (-,-,-).

Cada arco representa um disparo $M_1\ [b\rangle\ M_2$ onde M_1 e M_2 são marcações alcançáveis, enquanto **b** é um elemento de amarração em M_1. Adotou-se **EM,i** e **RM,i,k** ao invés de (**EM**,$\langle r = g_i \rangle$) e (**RM**,$\langle r = g_i, d = g_k \rangle$). Representação análoga foi adotada para **RT** e **ER**.

Deve-se observar que, em contraste com a teoria dos grafos, é permitido que um grafo de disparos tenha vários arcos entre o mesmo par ordenado de nós (por esta razão **A** é definido como um conjunto separado e não como um subconjunto de $V \times V$).

Diz-se que um arco **a** com $N(a) = (v_1,v_2)$ vai do **nó origem** v_1 ao **nó destino** v_2, o que define duas funções: $o,d \in [A \rightarrow V]$. Para cada arco, a primeira função mapeia para o seu nó origem, enquanto a segunda mapeia para o seu nó destino.

Definição 6.12: **Grafo de Disparos Integral.**
Um **grafo de disparos integral** de uma RPC é um grafo orientado $GD = (V,A,N)$, onde:

i) $V = [M_0\rangle$ é o conjunto de vértices;

ii) $A = \{(M_1,b,M_2) \in V \times EA \times V \mid M_1\ [b\rangle\ M_2\}$ é o conjunto de arestas;

iii) $\forall a = (M_1,b,M_2) \in A$: $N(a) = (M_1,M_2)$ é o conjunto de nós.

Quando todas as variáveis de uma RPC têm tipos finitos é direto provar que o **GD** é finito se e somente se todos os lugares são limitados.

Quando desenha-se **GD** como o da figura 6.3, usualmente escreve-se dentro de cada nó um texto descrevendo a marcação que o mesmo representa. Analogamente, escreve-se próximo a cada arco o elemento de amarração que ele representa.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

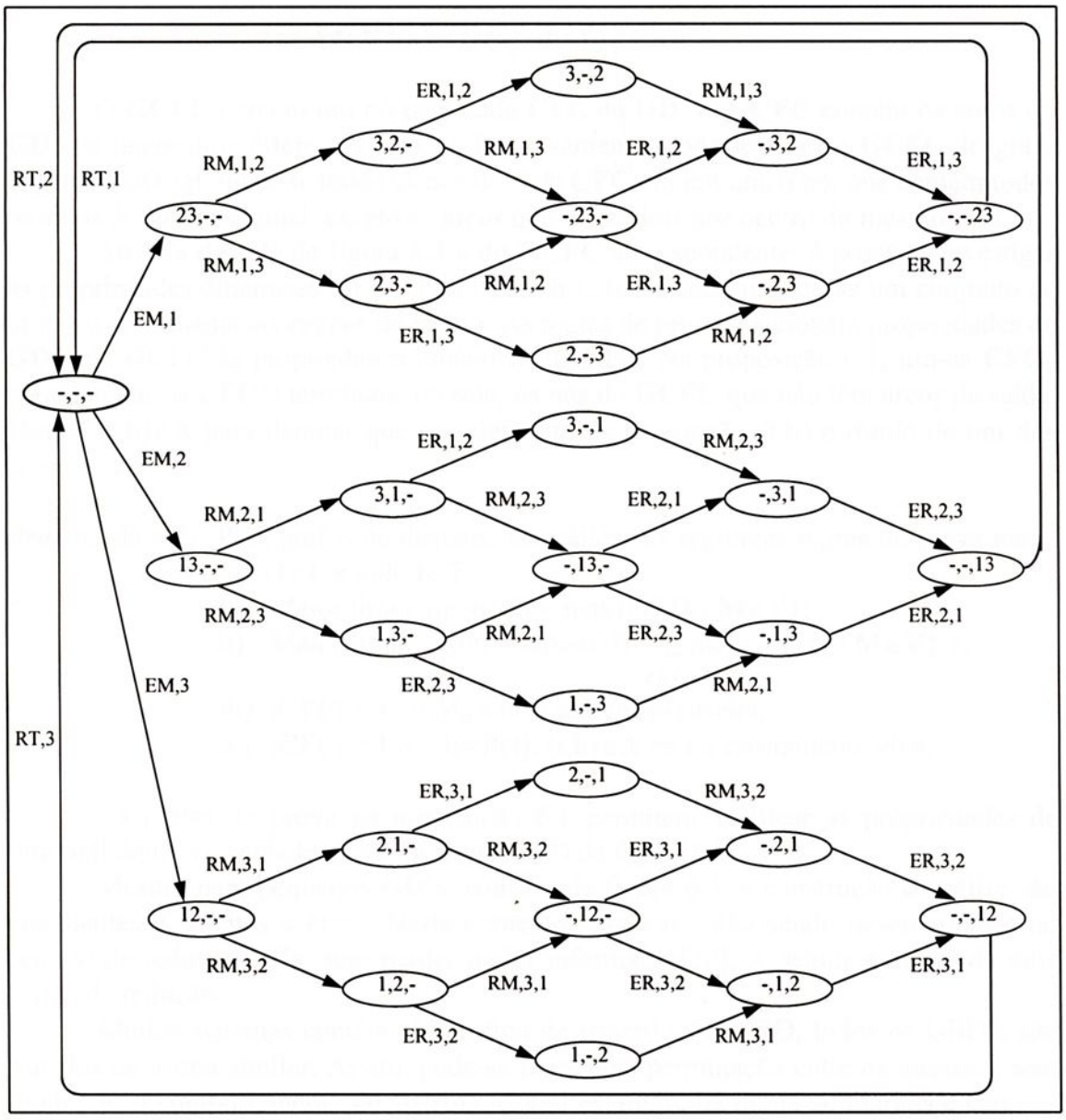


Fig. 6.3. Grafo de Disparos para o Sistema de Banco de Dados com 3 Gerenciadores.

Definem-se caminhos orientados finitos e componentes fortemente conexos de forma usual. Um componente fortemente conexo (**CFC**) é um subgrafo no qual existe um caminho orientado entre dois nós quaisquer. Maiores informações podem ser encontradas no apêndice.

Usa-se $\mathbf{v}^{\mathbf{c}}$ para denotar o **CFC** que contém **v**. Uma notação similar é usada para arcos.

Definição 6.13: O grafo orientado $\mathbf{G^* = (V^*,A^*,N^*)}$ é o **grafo de componentes fortemente conexos** do grafo orientado $\mathbf{GCFC = (V^*,A^*,N^*)}$ se e somente se as seguintes propriedades forem satisfeitas:

i) $\mathbf{V^* = CFC}$ é o conjunto dos componentes fortemente conexos;

ii) $\mathbf{A^* = \{a \in A \mid o(a)^c \neq d(a)^c\}}$;

iii)$\forall \mathbf{a} \in \mathbf{A}^*$: $\mathbf{N}^*(\mathbf{a}) = (\mathbf{o}(\mathbf{a})^c, \mathbf{d}(\mathbf{a})^c)$.

O **GCFC** contém um nó para cada **CFC** de **GD**. O **GCFC** contém os arcos do **GD** que unem dois diferentes **CFC**'s. Intuitivamente, pode-se obter o **GCFC** do grafo original. Para tal, junta-se todos os nós de cada **CFC** em um único nó, que contém todos os arcos do grafo original, exceto os arcos que unem dois nós dentro do mesmo **CFC**.

Através do **GD** da figura 6.3 e do **GCFC** correspondente, é possível investigar as propriedades dinâmicas do SBD da figura 6.1. Isto é feito usando-se um conjunto de proposições, chamadas **regras de prova**. As regras de prova relacionam propriedades de **GD** e de **GCFC** às propriedades dinâmicas da RPC. Na proposição 6.1, usa-se $\mathbf{CFC_T}$ para denotar os **CFC**'s terminais, ou seja, os nós do **GCFC** que não têm arcos de saída. Usa-se $(\mathbf{t},\mathbf{b}) \in \mathbf{A}$ para denotar que um elemento de amarração $(\mathbf{t},\mathbf{b})$ é rótulo de um dos arcos de **GD**.

Proposição 6.1: Para grafos de disparo, são válidas as seguintes regras de prova para todo $\mathbf{l} \in \mathbf{L}$ e todo $\mathbf{t} \in \mathbf{T}$:

i) Maior limite inteiro$(\mathbf{l}) = \max\{|\mathbf{M}(\mathbf{l})| : \mathbf{M} \in \mathbf{V}\}$;

ii) Maior limite multi-conjunto$(\mathbf{l}) = \sum_{\mathbf{c} \in \mathbf{C}(\mathbf{l})} \max\{\mathbf{M}(\mathbf{l},\mathbf{c}) : \mathbf{M} \in \mathbf{V}\}\text{'}\mathbf{c}$;

iii) $|\mathbf{CFC}| = 1 \Rightarrow \mathbf{M_0}$ é uma marcação caseira;

iv) $|\mathbf{CFC}| = 1 \wedge \forall\, \mathbf{b} \in \mathbf{B}(\mathbf{t})$: $(\mathbf{t},\mathbf{b}) \in \mathbf{A} \Rightarrow \mathbf{t}$ é estritamente viva.

As regras de prova na proposição 6.1 permitem verificar as propriedades de limitabilidade, vivacidade e caseira para o SBD da figura 6.1.

Mesmo para pequenos **GD**'s, como o da figura 6.3, a construção e análise são entediantes e sujeitas a erros. Neste contexto, técnicas estão sendo desenvolvidas no sentido de reduzir GD's, sem perder muita informação[20]. A seguir é esboçada uma forma de redução.

Muitos sistemas contém algum tipo de simetria. No SBD, todos os GBD's são tratados de forma similar. Assim, pode-se fazer uma permutação entre os mesmos, sem modificar o comportamento do sistema. Como exemplo, há muita similaridade entre as marcações (1,-,3) e (2,-,3). Estas marcações são simétricas, no sentido de que uma pode ser obtida a partir da outra através da permutação de $\mathbf{g_1}$ com $\mathbf{g_2}$.

Usa-se $\Phi_{\mathbf{GBD}}$ para denotar todas as permutações dos elementos do tipo **GBD**. Diz-se que $\phi \in \Phi_{\mathbf{GBD}}$ é uma **simetria** e que ela mapeia a marcação **M** em uma marcação simétrica $\phi(\mathbf{M})$, obtida de **M** pela troca de cada $\mathbf{g} \in \mathbf{GBD}$ por $\phi(\mathbf{g})$. Diz-se, também, que **M** e $\phi(\mathbf{M})$ são **equivalentes**, o que é escrito como $\mathbf{M} \approx_M \phi(\mathbf{M})$. $\Phi_{\mathbf{GBD}}$ é um grupo algébrico. Isto implica que $\approx_M$ é uma relação de equivalência. Notação e terminologia similares são usadas por elementos de amarração e para multi-conjuntos de valores de *tokens*. Todas as expressões de arcos do SBD (figura 6.1) satisfazem a seguinte propriedade para todo $\phi \in \Phi_{\mathbf{GBD}}$ e todo $\mathbf{b} \in \mathbf{B}(\mathbf{t})$:

$$(*) \quad \mathbf{E}(\mathbf{a})\langle\phi(\mathbf{b})\rangle = \phi(\mathbf{E}(\mathbf{a})\langle\mathbf{b}\rangle).$$

Intuitivamente, isto significa que amarrações simétricas têm efeitos simétricos. Pode-se obter os *tokens* usados pela amarração $\phi(\mathbf{b})$ aplicando-se ϕ aos *tokens* usados por **b**. A propriedade (*) é local e estática, podendo ser determinada de transições individuais sem se considerar o conjunto de todas as marcações alcançáveis. De (*)

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

pode-se provar a propriedade dinâmica a seguir, satisfeita para todas as marcações **M'**, **M"**$\in[\mathbf{M_0}>$, todos os elementos de amarração **b**$\in$**EA** e toda $\phi\in\Phi_{\mathbf{GBD}}$:

$$(**) \quad \mathbf{M'[b>\ M"} \Leftrightarrow \phi(\mathbf{M'})[\phi(\mathbf{b})>\phi(\mathbf{M"}).$$

Intuitivamente, isto significa que marcações simétricas têm elementos de amarração habilitados simétricos, os quais levam a marcações sucessoras simétricas. Quando (**) é satisfeita, faz sentido construir-se grafos de disparos onde tem-se somente um nó para cada classe de equivalência de marcações e um arco para cada classe de equivalência de elementos de amarração disparados. Tal grafo é denominado **grafo de disparos com simetria** (**GDS**). O **GDS** para o SBD com três GBD's é mostrado na figura 6.4, onde cada classe de equivalência é representada por meio de um de seus membros.

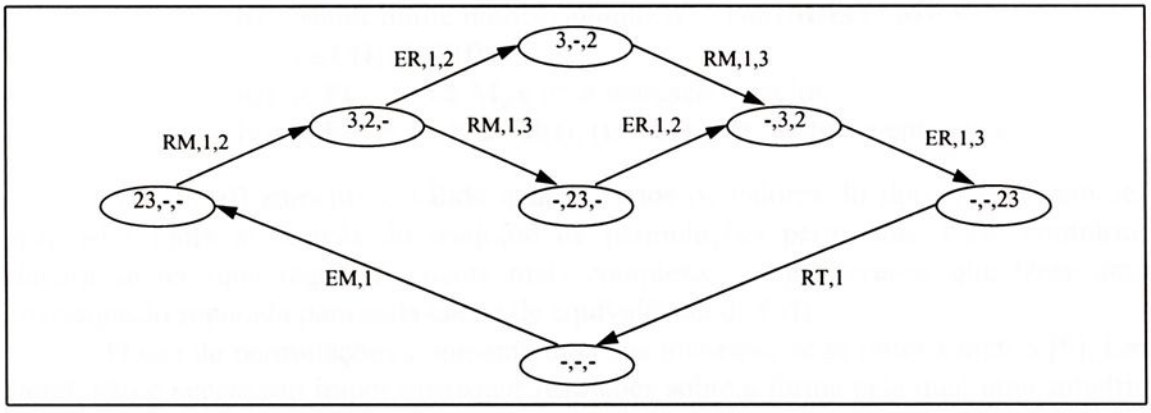


Fig. 6.4. Grafo de Disparo com Simetria para o SBD com 3 Gerenciadores.

Geralmente, um **GDS** é muito menor do que o **GD** correspondente. Para o SBD, o **GD** cresce com velocidade exponencial enquanto o **GDS** cresce quadraticamente [16]. A figura 6.5 ilustra a complexidade de **GD** e **GDS**. Entretanto, é igualmente importante o tempo gasto para calcular ambos os grafos. A complexidade do tempo de construção de **GD** é da ordem $O(\mathbf{n}^2 * 3^{\mathbf{n}})$, enquanto a complexidade do tempo de construção de **GDS** é da ordem $O(\mathbf{n}^3)$.

|GBD|	Grafo de Disparos Integral		Grafo de Disparos com Simetria	
O(n)	Nós - $O(n*3^n)$	Arcos - $O(n^2*3^n)$	Nós - $O(n^2)$	Arcos - $O(n^2)$
2	7	8	4	4
3	28	42	7	8
4	109	224	11	14
5	406	1090	16	22
6	1459	4872	22	32
7	5104	20426	29	44
8	17497	81664	37	58
9	59050	314946	46	74
10	196831	1181000	56	92
15	71744536	669615690	121	212
20	23245229341	294439571680	211	382

Fig. 6.5. O Tamanho dos Grafos de Disparo Integral e com Simetria para o SBD.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Embora os **GDS**'s sejam geralmente muito menores do que os correspondentes **GD**'s, eles são igualmente poderosos no que diz respeito a verificação de propriedades dinâmicas. As regras de prova na proposição 6.2 são intimamente relacionadas àquelas da proposição 6.1. Usa-se * para denotar a multiplicação de um multi-conjunto **GBD** por um inteiro. Como antes, usa-se $(\mathbf{t},\mathbf{b})\in\mathbf{A}$ para denotar que um elemento de amarração $(\mathbf{t},\mathbf{b})$ encontra-se em um dos arcos do grafo. Entretanto, deve-se observar que cada arco agora representa uma classe de equivalência de elementos de amarração. Assim, exige-se que $(\mathbf{t},\mathbf{b})$ pertença a uma dessas classes de equivalência.

Proposição 2.2: Para grafos de disparo com simetria, são válidas as seguintes regras de prova para todo $\mathbf{l}\in\mathbf{L}$ e todo $\mathbf{t}\in\mathbf{T}$:

i) Maior limite inteiro(**l**) = $\max\{|\mathbf{M}(\mathbf{l})| : \mathbf{M}\in\mathbf{V}\}$;

ii) Maior limite multi-conjunto(**l**) = $\max\{\mathbf{M}(\mathbf{l},\mathbf{c}): \mathbf{M}\in\mathbf{V} \wedge \mathbf{c}\in\mathbf{C}(\mathbf{l})\}^* \mathbf{C}(\mathbf{l})$;

iii) $|\mathbf{CFC}| = 1 \Rightarrow \mathbf{M_0}$ é uma marcação caseira;

iv) $|\mathbf{CFC}| = 1 \wedge \forall\, \mathbf{b}\in\mathbf{B}(\mathbf{t}): (\mathbf{t},\mathbf{b})\in\mathbf{A} \Rightarrow \mathbf{t}$ é estritamente viva.

A regra ii) somente é válida quando todos os valores do tipo **C**(**l**) podem ser mapeados entre si através do conjunto de permutações permitidas. Caso contrário, deveria-se ter uma regra levemente mais complexa, porque teria-se que fazer uma investigação separada para cada classe de equivalência de **C**(**l**).

O uso de permutações é somente uma das maneiras de se obter simetria [9]. Em geral, não é necessário impor quaisquer restrições sobre a forma pela qual uma simetria mapeia marcações em marcações e elementos de amarração em elementos de amarração, bastando satisfazer (**).

Há muitas outras formas de reduzir grafos de disparos. **Conjuntos inflexíveis** descartam alguns dos muitos elementos de amarração [12,13]. **Marcações cobertas** cuidam das situações nas quais alguns lugares tornam-se ilimitados [14]. Acredita-se que seja possível combinar o uso de simetria, conjuntos inflexíveis e marcações cobertas [15], o que seria melhor.

A análise através de grafos de disparos tem várias propriedades atrativas. Primeiro, é extremamente fácil usar grafos de disparo. A construção e análise pode ser completamente computadorizada. Isto significa que o modelador não necessita ter qualquer conhecimento da matemática subjacente. Por exemplo, não é necessário saber como checar se duas marcações são simétricas ou não, ou ainda, como trabalham as regras de prova. Segundo, o grafo de disparos contém todos os detalhes a respeito do comportamento da RPC, uma vez que ele representa todas as possíveis seqüências de disparos. Assim, é possível investigar todos os tipos de propriedades dinâmicas por meio de grafos de disparos.

Um problema com grafos de disparos é o fato de ser necessário fixar todos os parâmetros do sistema (como, por exemplo, o número de GBD's no SBD) antes de sua construção. Isto significa que sempre serão encontradas propriedades que são específicas aos valores escolhidos para os parâmetros do sistema. Na prática, o problema não é tão grande. O entendimento de como um SBD trabalha com poucos GBD's leva ao entendimento de como o sistema trabalha para muitos GBD's. Naturalmente, isto somente é verdade quando da discussão sobre a validação do modelo e não quando se fala sobre o seu desempenho.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

PARTE II:
REDES DE PETRI TEMPORIZADAS DE ALTO-NÍVEL

unesp

CAPÍTULO 7: DESENVOLVIMENTO TEÓRICO DE RPTAN

7.1. INTRODUÇÃO

Conforme destacado na introdução desta dissertação de mestrado, a motivação deste trabalho nasceu da necessidade de simular, em um computador de pequeno porte, o comportamento dos componentes de uma arquitetura computacional dedicada à simulação de circuitos eletrônicos, altamente paralela e com fortes restrições temporais, proposta por N.Marranghello e F.Damiani (arquitetura ABACUS [24]). Para a construção do modelo da arquitetura em questão, inicialmente, adotou-se as Redes de Petri devido à sua potencialidade gráfica e facilidade de compreensão por terceiros.

B.C.Damasceno et al. [7] demonstraram que uma RP clássica é pouco adequada no tratamento de questões temporais e de estruturas repetitivas, presentes no modelo da arquitetura ABACUS.

Assim, partiu-se para o estudo de diferentes extensões de RP, procurando encontrar dentre elas elementos que pudessem resolver os problemas de modelagem apresentados pela RP clássica. As RP Temporizadas de Merlin [5] mostraram-se adequadas para modelar os aspectos temporais da arquitetura ABACUS, enquanto as RP Numéricas [10] mostraram-se adequadas no tratamento das estruturas repetitivas. Entretanto, inexistia um simulador de RP para PC que unisse as características de ambas as redes. Desta forma, a partir das principais características operacionais das redes citadas, com a adição de outras características, formou-se a base de uma nova extensão de RP, a qual foi denominada Rede de Petri Numérico-Temporizada (RPNT). A partir de um conjunto de definições iniciais, construiu-se um simulador para a nova extensão, o qual faz parte de um trabalho de iniciação científica (processo FAPESP 93/4986-2) [26,27].

Entretanto, durante o desenvolvimento de exemplos de modelos em RPNT, observou-se a necessidade de mudanças na caracterização dessa extensão, visando uma utilização mais fácil e direta no que dizia respeito ao processo de criação de um modelo de sistema. Assim, no presente trabalho de mestrado, fez-se um estudo mais aprofundado dos principais tipos de RP, procurando subsídios que pudessem melhorar significativamente a representação e forma de funcionamento das RPNT. A partir deste estudo, encontrou-se nas RP Coloridas a representação mais completa e flexível dentre as diferentes extensões de RP. Desta forma, procurou-se transferir para as RPNT os princípios que caracterizam a potencialidade que as RP Coloridas têm em trabalhar sobre *tokens* diferenciáveis, gerando, assim, uma segunda versão de RPNT. Devido às suas características de RP de alto-nível, houve-se por bem denominar a segunda versão de RPNT por Rede de Petri Temporizada de Alto-Nível (RPTAN).

As RPTAN são tratadas nas seções seguintes. A seção 7.2 traz a representação e forma de funcionamento desta extensão. A definição de RPTAN é fornecida na seção 7.3. A seção 7.4 fornece um exemplo de modelagem em RPTAN. Outros exemplos de modelagem encontram-se no capítulo 9.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

7.2. Representação e Regras de Funcionamento

A figura 7.1 serve para dar uma primeira ilustração sobre RPTAN.

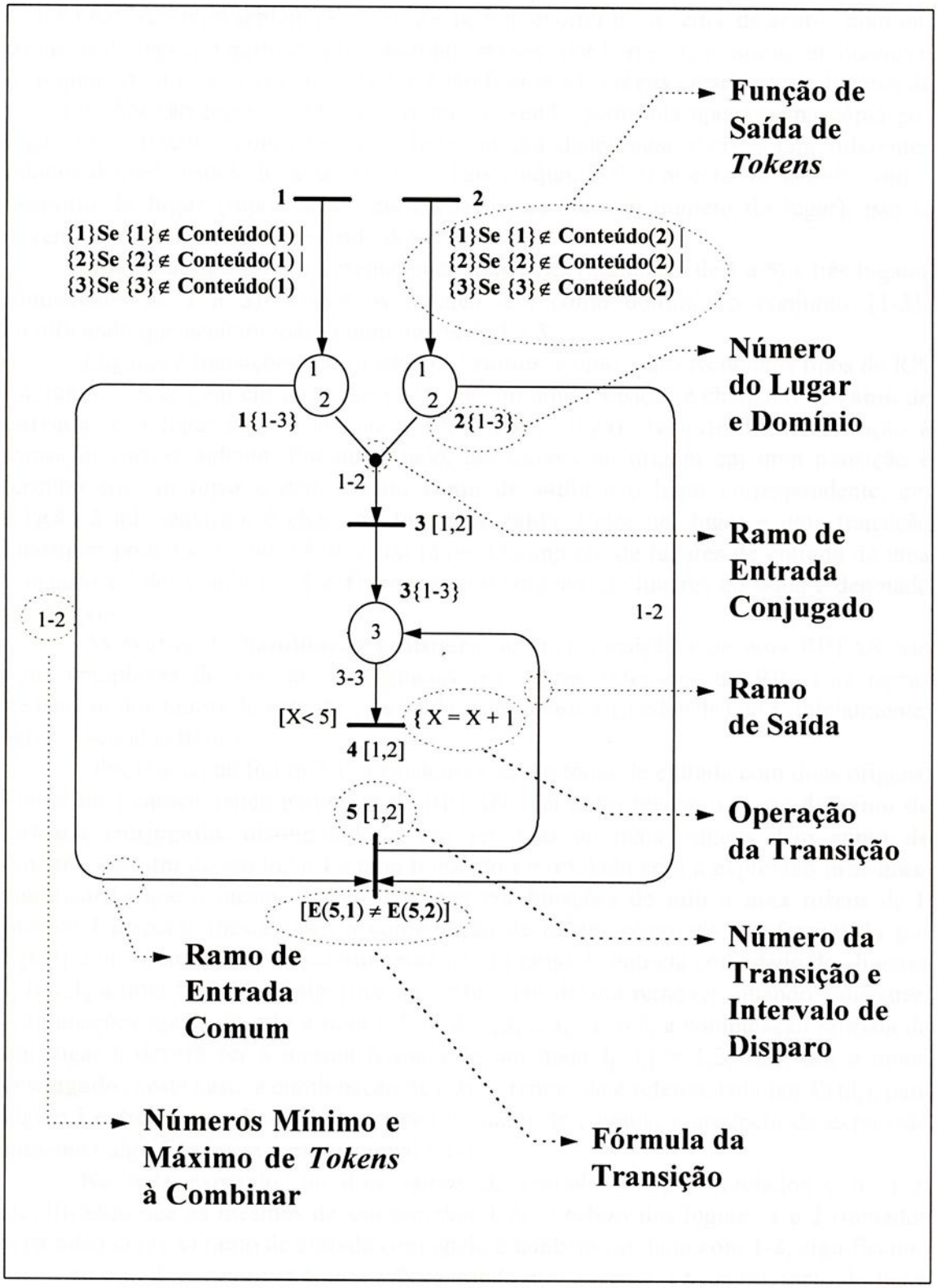


Fig. 7.1. Exemplo de uma RPTAN.

Tal como as demais extensões de RP, RPTAN pode ser apresentada como um grafo que representa o estado de um sistema e as ações que podem ocorrer nele, promovendo mudanças de estado. Os **lugares** de uma RPTAN representam os estados locais do sistema modelado. Graficamente, são representados por elipses. As **transições** de uma RPTAN representam as ações que podem ocorrer no sistema, de acordo com um conjunto de regras. Graficamente, são representadas por barras. O conjunto de lugares e o conjunto de transições de uma RPTAN são finitos. Os ***tokens*** presentes nos lugares de uma RPTAN são representados por números, sendo permitida apenas uma cópia por lugar. Os diferentes conjuntos de *tokens* em um dado lugar representam diferentes estados daquele estado local do sistema. Tais conjuntos devem estar de acordo com o **domínio do lugar** (representado entre chaves, ao lado do número do lugar), isto é, devem ser subconjuntos do referido domínio.

A rede da figura 7.1 apresenta cinco transições (numerada de **1** a **5**) e três lugares (numerados de **1** a **3**). Todos os lugares têm como domínio o conjunto {**1-3**}, significando que aceitam *tokens* numerados de **1** a **3**.

Lugares e transições são ligados por **ramos**. Como todos os demais tipos de RP, um ramo com origem em um lugar e término em uma transição é chamado de **ramo de entrada**, e o lugar correspondente é chamado de **lugar de entrada**, em relação à transição correspondente. Por outro lado, um ramo com origem em uma transição e término em um lugar é denominado **ramo de saída** e o lugar correspondente, em relação à tal transição, é chamado **lugar de saída**. Entre um lugar e uma transição quaisquer pode existir no máximo um ramo. O conjunto de lugares de entrada de uma transição **t** é designado por **LE(t)**, enquanto o conjunto de lugares de saída é denotado por **LS(t)**.

As **regras de habilitação e disparo** de uma transição **t** de uma RPTAN são mais complexas do que as apresentadas por outras extensões de RP. Tais regras dependem dos ramos de entrada e de saída e da **fórmula** (predicado) de **t**. Inicialmente, serão discutidos ramos.

Observa-se, na figura 7.1, a existência de um ramo de entrada com duas origens. Esta é uma característica particular de RPTAN. Tal ramo recebe o nome de **ramo de entrada conjugado**, distinguindo-se por ter duas ou mais origens. Um **ramo de entrada comum** de um lugar **l** a uma transição **t** é rotulado com a expressão **min-max**, significando que o mesmo poderá remover combinações de **min** a **max** *tokens* de **l**, quando **t** disparar (neste caso, a combinação de *tokens* removida é referenciada por **E(t,l)**). Por outro lado, o rótulo **min-max** de um ramo de entrada conjugado dos lugares $\mathbf{l_1},\mathbf{l_2},...,\mathbf{l_n}$ a uma transição **t** significa que este ramo deverá remover, quando **t** disparar, combinações iguais de **min** a **max** *tokens* de $\mathbf{l_1},\mathbf{l_2},...,\mathbf{l_n}$, isto é, a combinação retirada de um lugar $\mathbf{l_i}$ deverá ser a mesma retirada de um lugar $\mathbf{l_j}$, $\mathbf{i,j} = 1,2,...,\mathbf{n}$, daí o nome **conjugado** (neste caso, a combinação de *tokens* removida é referenciada por $\mathbf{E(t,l_i)}$, para algum **i** entre 1 e **n**). Para os dois tipos de ramo de entrada, a ausência da expressão **min-max** significa que a mesma é igual a **1-1**.

Na rede exemplo, há dois ramos de entrada comuns rotulados com **1-2**, significando que os mesmos devem remover 1 ou 2 *tokens* dos lugares **1** e **2** (tomados separadamente). O ramo de entrada conjugado é também rotulado com **1-2**, significando que o mesmo deve remover 1 ou 2 *tokens*, sendo o(s) mesmo(s) *token*(*s*), tanto do lugar **1** quanto do lugar **2**.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Pode-se atribuir uma função **S(t,s)** para o ramo de saída ligando a transição **t** ao lugar **s**. **S(t,s)** é da forma:

Δ_1 **SE** exp_booleana $_1$ |
Δ_2 **SE** exp_booleana $_2$ |
...
Δ_k **SE** exp_booleana $_k$

onde exp_booleana$_i$ é do tipo **exp_booleana**, na qual pode-se utilizar constantes, variáveis, funções pré-definidas e combinações de *tokens* **E(t,l)**, para todo lugar de entrada **l** de **t**. Com o disparo de **t**, o conjunto de *tokens* Δ_i será repassado para o lugar **s** caso Expressão_Booleana$_i$ seja verdadeira, **i** = 1,...,**k**. Tão logo isto aconteça, as demais expressões serão desconsideradas. Deve-se ressaltar que $\Delta_i \cap$ **Conteúdo(s)**=$\varnothing$ é uma das condições de habilitação de **t**, onde a função **Conteúdo(s)** retorna o conjunto de *tokens* presentes em **s** no estado corrente da rede. Caso não explicitada, **S(t,s)** equivale a união dos combinações de *tokens* retiradas dos lugares de entrada de **t**.

A outra regra de habilitação de uma transição **t** refere-se à **fórmula da transição**. A fórmula da transição também é do tipo **exp_booleana**, a qual deve ser satisfeita para que a transição possa ser disparada.

Na rede da figura 7.1, a transição **5** tem por fórmula [**E(5,1)** $\neq$ **E(5,2)**]. Isto significa que tal transição é desabilitada caso tente-se remover combinações de *tokens* iguais dos lugares **1** e **2**, uma vez que a fórmula resultaria no valor booleano **Falso**. Para combinações distintas de *tokens* retiradas dos lugares **1** e **2**, a fórmula da transição resultará no valor booleano **Verdadeiro**, o que é uma das condições de habilitação da transição **5**.

Conforme descrito anteriormente, **E(t,l)** denota a combinação de *tokens* **removida** do lugar **l** mediante o disparo de **t**. De acordo com os *tokens* presentes em **l** num dado estado da rede, **E(t,l)** deve ser escolhida dentre uma classe constituída pelas combinações de *tokens* que podem ser formadas respeitando-se o rótulo **min-max** do ramo de entrada (comum ou conjugado), classe esta denotada por **CE(t,l)**. Desta forma, para todo **l**$\in$**LE(t)**, a transição **t** poderá estar habilitada em relação a alguns elementos de **CE(t,l)** e não estar habilitada para outros.

Destarte, para que uma transição **t** torne-se habilitada, as seguintes condições devem ser satisfeitas:

a) $\forall$**l**$\in$**LE(t)**, a fórmula da transição deve ser satisfeita para algum elemento (combinação) de **CE(t,l)**. Caso ausente, a fórmula corresponde ao valor booleano **Verdadeiro**; e
b) $\forall$**s**$\in$**LS(t)**, **S(t,s)** $\cap$ **Conteúdo(s)** = $\varnothing$.

Chama-se **amarração** de **t** a tupla (**t**, **E(t,l$_1$)**=Ω_1, **E(t,l$_2$)**=Ω_2, ..., **E(t,l$_n$)**=Ω_n), onde $\cup_{i=1,...,n}$ **l$_i$** = **LE(t)**, com $\Omega_i \in$ **CE(t,l$_i$)**, **i** = 1,...,**n**.

Assim, uma transição **t**, em um dado estado da rede, pode estar habilitada para uma amarração e não estar habilitada para outra.

Na rede da figura 7.1, tem-se:

CE(3,1) = **CE(3,2)** = {{1},{2},{3},{1,2},{1,3},{2,3}}
CE(4,3) = {{1,2,3}}
CE(5,1) = {{1},{2},{3},{1,2},{1,3},{2,3}}

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

CE(5,2) = {{1},{2},{3},{1,2},{1,3},{2,3}}
[E(5,1) ≠ E(5,2)]
LS(1) = {1}
LS(2) = {2}
LS(3) = {3}
LS(4) = ∅
LS(5) = {3}
Conteúdo(1) = {1,2}
Conteúdo(2) = {1,2}
Conteúdo(3) = {3}

Desta forma, na figura 7.1, de acordo com as condições de habilitação, as transições **1** e **2** só podem ser habilitadas para a inserção do conjunto de *tokens* {3} nos lugares **1** e **2**, respectivamente. A transição **3** pode ser habilitada para as seguintes amarrações:

(**3**, **E(3,1)** = **E(3,2)** = {1})
(**3**, **E(3,1)** = **E(3,2)** = {2})
(**3**, **E(3,1)** = **E(3,2)** = {1,2})

Por outro lado, a transição **4** não encontra-se habilitada, pois necessita da presença de 3 *tokens* no lugar **3**. Finalmente, a transição **5** pode ser habilitada para as amarrações:

(**5**, **E(5,1)** = {1}, **E(5,2)** = {2})
(**5**, **E(5,1)** = {1}, **E(5,2)** = {1,2})
(**5**, **E(5,1)** = {2}, **E(5,2)** = {1})
(**5**, **E(5,1)** = {2}, **E(5,2)** = {1,2})
(**5**, **E(5,1)** = {1,2}, **E(5,2)** = {1})
(**5**, **E(5,1)** = {1,2}, **E(5,2)** = {2})

Para que possa disparar, uma transição deve estar habilitada e respeitar sua **regra de disparo**, que é uma condição referente ao **tempo**.

Tal como as RP Temporizadas de Merlin, RPTAN utilizam-se de **intervalos de disparo** associados às transições. Chama-se **intervalo de disparo estático** aquele que é anotado junto ao número da transição, entre colchetes (caso ausente, assume-se [0,0]). Este intervalo é adotado tão logo a transição torne-se habilitada. A partir deste instante, o intervalo passa a ser chamado de **intervalo de disparo dinâmico**. O intervalo estático corresponde ao valor *default* do intervalo de disparo, sendo denotado por [**LDI**,**LDS**], onde **LDI** é o **limite de disparo inferior** e **LDS** é o **limite de disparo superior**. O intervalo dinâmico é decorrente do decréscimo de tempo sobre o estático no período de tempo em que a transição continua habilitada sem disparar. O conjunto dos intervalos de disparo de um conjunto de transições habilitadas pode ser representado por um gráfico de interseções, como o da figura 7.2. Observando essa figura, são as seguintes as regras de disparo de uma transição:

a) O limite superior do intervalo de disparo dinâmico deve ser igual à 0; ou
b) O limite inferior do intervalo de disparo dinâmico precisa ser menor ou igual ao mínimo dos limites superiores dos intervalos de disparo dinâmicos das transições habilitadas;

unesp

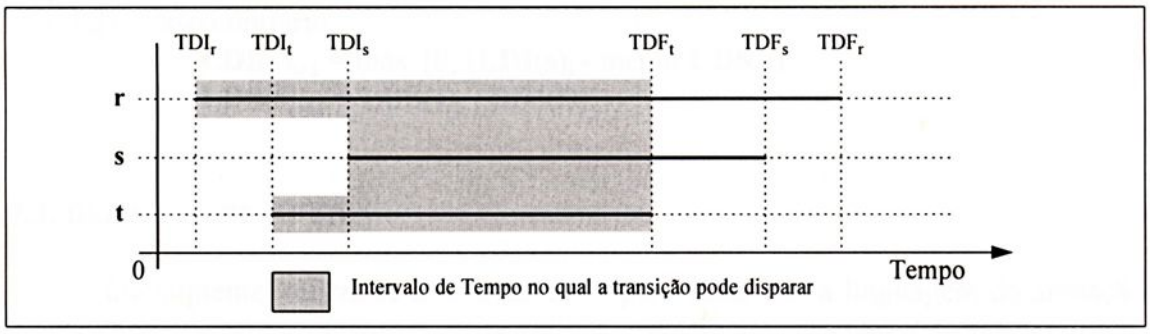


Fig. 7.2. Gráfico de Interseções dos Intervalos de Disparo.

Antes de tratar-se dos efeitos de um disparo, deve-se destacar que os estados de uma RPTAN são chamados **classes**. Uma classe $\mathbf{C_i}$ ($\mathbf{i} \in \mathbf{IN}$) é constituída de três conjuntos, $\mathbf{C_i} = (\mathbf{M_i}, \mathbf{D_i}, \mathbf{V_i})$, onde:

a) $\mathbf{M_i}$: **marcação**;
b) $\mathbf{D_i}$: **domínio de tempo**;
c) $\mathbf{V_i}$: **variáveis**.

Caso a transição **t** dispare na classe $\mathbf{C_i}$, o efeito de tal disparo promoverá a mudança da classe atual para uma nova classe $\mathbf{C_{i+1}}$, o que ocorre em três etapas:

a) *Remoção de tokens dos lugares de entrada e adição de tokens aos lugares de saída.*
De cada lugar $\mathbf{l} \in \mathbf{LE(t)}$, será removida a combinação de *tokens* $\mathbf{E(t,l)}$. Para cada lugar $\mathbf{s} \in \mathbf{LS(t)}$, será adicionado o conjunto de *tokens* $\mathbf{S(t,s)}$. Caso ausente, $\mathbf{S(t,s)}$ assume o conjunto $\cup_{\mathbf{l} \in \mathbf{LE(t)}} \mathbf{E(t,l)}$.

b) *Realização das **operações da transição**.*
Estas operações consistem de atribuições do tipo **Variável ← Valor** e de comandos condicionais do tipo **IF-THEN-ELSE**. Tais operações (introduzidas por uma chave, próximo à transição) servem para alterar a memória da rede, ou seja, seu conjunto de variáveis; e

c) *Atualização do **tempo** da rede.*
Analisando-se o estado futuro da rede, são válidas as seguintes regras:
c.1) Se a transição que disparou permanecer habilitada em $\mathbf{C_{i+1}}$, então seu intervalo de disparo assumirá os valores do intervalo estático;
c.2) As transições que se tornaram habilitadas na classe alcançada $\mathbf{C_{i+1}}$ têm seus intervalos de disparo atualizados para seu valor *default* (estático);
c.3) Para cada transição **s** habilitada em $\mathbf{C_i}$ que permanecer habilitada em $\mathbf{C_{i+1}}$, o seu intervalo de disparo será calculado da seguinte forma:
c.3.1) Caso $\mathbf{i} > 0$ e as transições **t** e **s** estavam habilitadas em $\mathbf{C_{i-1}}$:

$$\mathbf{LDI(s)_{i+1}} = \text{máx}\ \{0;\ (\mathbf{LDI(s)_i} - \text{menor}\ \mathbf{LDS_i});\ (\mathbf{LDI(s)_{i-1}} - \mathbf{LDS(t)_{i-1}})\}$$

$$\mathbf{LDS(s)_{i+1}} = \min\ \{(\mathbf{LDS(s)_i} - \mathbf{LDI(t)_i});\ (\mathbf{LDS(s)_{i-1}} - \mathbf{LDI(t)_{i-1}})\}$$

onde:

$\mathbf{LDI(s)_i}$ é o limite de disparo inferior de **s** em $\mathbf{C_i}$;
$\mathbf{LDS(s)_i}$ é o limite de disparo superior de **s** em $\mathbf{C_i}$;
$\mathbf{LDS_i}$ é o conjunto dos limites de disparo superiores das transições habilitadas em $\mathbf{C_i}$.

c.3.2) Caso contrário:

$\mathbf{LDI(s)_{i+1}}$ = máx {0; ($\mathbf{LDI(s)_i}$ - menor $\mathbf{LDS_i}$)}

$\mathbf{LDS(s)_{i+1}}$ = $\mathbf{LDS(s)_i}$ - $\mathbf{LDI(t)_i}$.

7.3. DEFINIÇÃO DE RPTAN

Inicialmente, utiliza-se a notação BNF para descrever a linguagem de anotação de RPTAN. Nesta descrição, **< constante >** e **< variável >** seguem a definição padrão de linguagens de programação.

< exp_booleana > ::= Verdadeiro | Falso |
< símb_relação > (< termo >) |
(< termo > < símb_relação_binária > < termo >) |
(< exp_booleana > AND < exp_booleana >) |
(< exp_booleana > OR < exp_booleana >) |
(NOT< exp_booleana >)

< símb_relação > ::= **Lugar_Vazio |** (outros, de acordo com o modelo)

< símb_relação_binária > ::= $\in | \notin | \subset | \subseteq | \not\subset | = | \neq |$
$< | \leq | > | \geq |$ (outros, de acordo com o modelo)

< termo > ::= < constante > | < variável > |
< símb_função > (< termo >) |
(< termo > < símb_função_binária > < termo >) |
{< termo >, ...}

< símb_função > ::= Conteúdo | (outros, de acordo com o modelo)

< símb_função binária > ::= + | - | * | / | (outros, de acordo com o modelo)

< atribuição > ::= < termo> = < termo > < símb_função_binária > < termo >

< fórmula > ::= < exp_booleana >

< operação > ::= < atribuição > |
IF < fórmula > THEN < atribuição >* ELSE < atribuição >*

< função_saída_de_tokens > ::= N/A |
(< termo > SE < exp_booleana > |) *

Uma RPTAN é representada pela tupla ($\mathbf{L,T,RE,RC,RS,DL,F,O,D,C_0}$) , onde:

- **$L \subset IN^*$, contínuo, finito, não vazio - conjunto de lugares**
- **$T \subset IN^*$, contínuo, finito, não vazio - conjunto de transições**
- **RE: $T \times L \rightarrow IN^2$ - função ramos de entrada**

unesp

$$RE(t, l) = \begin{cases} (0,0), \text{ se inexistir ramo de tal tipo entre l e t;} \\ (a,b),\ 0 < a \leq b, \text{ se existir ramo de tal tipo entre l e t, com rótulo a - b.} \end{cases}$$

- **RC: $T \times L^n \rightarrow IN^2$ - função ramos de entrada conjugados**

$$RC(t, l_1,...,l_n) = \begin{cases} (0,0), \text{ se inexistir ramo de tal tipo entre } l_1,...,l_n \text{ e t;} \\ (a,b),\ 0 < a \leq b, \text{ se existir ramo de tal tipo entre } l_1,...,l_n \text{ e t, com rótulo a - b.} \end{cases}$$

- ***Saída*** $= \{ x \mid x = <$ função_saída_de_tokens $> \}$
  RS: $T \times L \rightarrow$ *Saída* - função ramos de saída

$$RS(t, l) = \begin{cases} N/A, \text{ se inexistir ramo de tal tipo entre t e l;} \\ x \neq N/A, \text{ se existir ramo de tal tipo entre t e l, sem função de saída de tokens.} \end{cases}$$

- **DL: $L \rightarrow IP(IN^*) \setminus \varnothing$ - função domínios dos lugares**
  DL (l) = X, limitado
- ***Fórmulas*** $= \{ x \mid x = <$ fórmula $> \}$
  F: $T \rightarrow$ *Fórmulas* - função fórmulas das transições
  F (t) = f
- ***Operações*** $= \{ x \mid x = <$ operação $> \}$
  O: $T \rightarrow$ *Operações* - função operações das transições
  O (t) = o
- **D: $T \rightarrow IR^+ \times IR^+$ - função intervalos de disparo**
  D (t) = (a, b), $a \leq b \leq \infty$
- **$C_0 = (M_0, D_0, V_0)$ - classe inicial**
- **$M_0 : L \rightarrow IP(DL(L))$ - função marcação inicial**
  $M_0(l) = X \in IP(DL(l))$
- **$D_0 = D$ - função domínio de tempo inicial**
- ***Variáveis*** $= \{ x \mid x = <$ variável $> \}$
  V_0 : *Variáveis* $\rightarrow$ IR - função valores iniciais das variáveis
  $V_0(v) = r$

Descrição dos componentes de uma RPTAN:

a) **Domínio do Lugar** - Conjunto de números naturais, representando os tipos de *tokens* que podem ocupar um dado lugar. Caso ausente, assume-se {0};

b) **Tokens Iniciais do Lugar** - Subconjunto do domínio do lugar, correspondendo aos *tokens* presentes no lugar quando a rede é inicializada (marcação $\mathbf{M_0}$ da classe $\mathbf{C_0}$);

c) **Variáveis da Rede** - Conjunto de variáveis que podem ser utilizadas por fórmulas, atribuições, comandos condicionais e funções;

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

d) **Fórmula da Transição** - Expressão booleana referente aos dados na memória da rede e às combinações de *tokens* provenientes dos ramos de entrada comuns e conjugados da transição. Em tempo de simulação, a fórmula da transição assume o valor booleano **Verdadeiro**, se resultar em um valor diferente de 0, e o valor booleano **Falso**, caso contrário;

e) **Intervalo de Disparo** - Intervalo de números reais não negativos, representando os tempos mínimo e máximo em que uma transição habilitada deve disparar. Chama-se **intervalo de disparo estático** aquele que a transição assume imediatamente após sua habilitação. Caso ausente, assume-se [0,0]. Por outro lado, chama-se **intervalo de disparo dinâmico**, o intervalo de disparo, num dado momento de simulação, de uma transição já habilitada (este intervalo varia caso outra transição habilitada dispare);

f) **Operações da Transição** - Conjunto de atribuições e comandos condicionais referentes às variáveis da rede, executados imediatamente após o disparo da transição;

g) **Ramos de Entrada** - Ramo com origem num lugar e término numa transição. Associa-se a tal ramo um intervalo de inteiros positivos **min-max**, indicando os números mínimo e máximo de *tokens* a formar a combinação a ser removida do lugar quando do disparo da transição. A combinação de *tokens* removida é designada por **E(t,l)**, onde **l** e **t** são respectivamente o lugar e a transição ligados pelo ramo de entrada;

h) **Ramos de Entrada Conjugados** - Ramo com origem em dois ou mais lugares e término numa transição. Associa-se a tal ramo um intervalo de inteiros positivos **min-max**, indicando os números mínimo e máximo de *tokens* a formar a combinação a ser removida por igual de cada um dos lugares quando do disparo da transição. A combinação de *tokens* removida é designada por **E(t,l)**, onde **l** deve ser algum dos lugares ligados a transição **t** pelo ramo de entrada conjugado;

i) **Ramos de Saída** - Ramo com origem numa transição e término num lugar. Associa-se a tal ramo uma **função de saída de *tokens***, indicando qual conjunto de *tokens* deve ser inserido no lugar quando do disparo da transição. Caso esta função esteja ausente, o conjunto formado pela união das combinações de *tokens* retiradas dos lugares de entrada da transição será transportado para o lugar de saída respectivo, quando do disparo da transição;

j) **Marcação** - É um conjunto de tuplas, onde o primeiro elemento de cada tupla é um número indicando um lugar e os demais elementos são números representando os *tokens* presentes no respectivo lugar, em determinado momento. Lugares vazios não são representados;

k) **Domínio de Tempo** - Conjunto de triplas, onde o primeiro elemento de cada tripla é um número representando uma transição habilitada e os demais elementos representam os limites inferior e superior do intervalo de disparo dinâmico da respectiva transição numa dada marcação da rede; e

l) **Classe Inicial** - É uma tripla $\mathbf{C_0} = (\mathbf{M_0}, \mathbf{D_0}, \mathbf{V_0})$, onde $\mathbf{M_0}$ é a **marcação inicial** da rede, $\mathbf{D_0}$ o **domínio de tempo inicial** e $\mathbf{V_0}$ é a **lista de variáveis** e seus respectivos valores iniciais.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Nas RPTAN, as seguintes extensões foram feitas às RP Lugar/Transição:

a) *Tokens* são **distinguíveis**. Assim, ao invés de serem representados pelo símbolo ●, são representados por números naturais;

b) Uma **memória** de escrita e leitura é associada à rede. Tal memória é representada por um conjunto de variáveis;

c) O conceito de marcação é ampliado para o de **classe**. A classe de uma RPTAN é dada por $\mathbf{C_i} = (\mathbf{M_i},\mathbf{D_i},\mathbf{V_i})$, onde $\mathbf{M_i}$ é uma marcação denotando **quais** os *tokens* presentes nos lugares da rede, $\mathbf{D_i}$ é o domínio dos **intervalos de disparo dinâmicos** associados às transições e $\mathbf{V_i}$ é um conjunto de **variáveis** e seus respectivos valores;

d) As regras de habilitação de uma transição **t**, além de envolverem os lugares de entrada e de saída da mesma, envolvem também uma **fórmula** e um intervalo de disparo associados a **t**;

e) O disparo de uma transição **t** promove a remoção de uma combinação α de *tokens* de seus lugares de entrada (de acordo com os limites dados pelos ramos de entrada e ramos de entrada conjugados de **t**), a adição de *tokens* em seus lugares de saída (de acordo com as **funções de saída de *tokens*** dos ramos de saída), a realização de **operações** sobre a memória da rede (operações da transição) e a alteração dos intervalos de disparo de transições.

O **Grafo de Classes** de uma RPTAN é um grafo que mostra todas as classes acessíveis na rede, a partir de uma dada classe inicial. Nas RPTAN, cada nó desse grafo representa uma classe acessível e cada arco direcionado é rotulado pela amarração de transição cujo disparo faz a rede passar da classe ligada na origem do arco à classe ligada no destino do arco. A amarração de transição é designada por uma tupla ordenada, onde o primeiro elemento é o número da transição e os demais elementos são as combinações de *tokens* que são removidas dos lugares de entrada quando do disparo da transição. Estas combinações são ordenadas entre si pela **ordem crescente** dos números dos lugares das quais são removidas. Ressalta-se que, em se tratando de ramos de entrada conjugados, a ordenação adota o menor número de lugar dentre os números dos lugares relacionados pelo ramo.

As propriedades de uma RPTAN são responsáveis pela validação ou não de propriedades do sistema modelado. São as seguintes as propriedades de RPTAN:

a) **Limitabilidade** - Uma RPTAN é **limitada** para uma dada classe inicial $\mathbf{C_0}$ se o seu grafo de classes for **finito**, o que ocorre somente se todas as variáveis usadas na rede assumirem um conjunto finito de valores. A validação dessa propriedade garante a possibilidade de implementação física do sistema;

b) **Reiniciabilidade** - Uma RPTAN limitada é **reiniciável** para uma dada classe inicial $\mathbf{C_0}$ se, para qualquer classe $\mathbf{C_n}$ decorrente da mesma, existir um caminho no grafo de classes com início em $\mathbf{C_n}$ e término em $\mathbf{C_0}$. A validação dessa propriedade assegura que o sistema modelado retorna ao seu estado inicial após realizar uma ou mais tarefas; e

unesp

c) **Vivacidade** - Uma RPTAN limitada é **viva** para uma dada classe inicial $\mathbf{C_0}$ se, para qualquer classe decorrente $\mathbf{C_n}$ e para uma transição **t** considerada, existir uma seqüência σ de disparos de transições, de forma que ao final dessa seqüência a transição **t** se encontre habilitada. A validação dessa propriedade assegura que o sistema correspondente está livre de *deadlocks* (impasses).

7.4. Exemplo de Modelagem em RPTAN

A figura 7.3 mostra uma RPTAN modelando um protocolo de comunicação que permite a transmissão de uma mensagem, dividida em 2 pacotes (representados pelos *tokens* **1** e **2**), entre duas entidades. Este protocolo é utilizado para transmitir mensagens de um lugar para outro, permitindo apenas a transmissão de um pacote de cada vez, sendo a comunicação bidirecional não simultânea (*half-duplex*). Hipóteses sobre o comportamento do meio de transmissão são que os pacotes ou reconhecimentos podem ser perdidos em trânsito. A recuperação de perdas é feita usando-se de retransmissões, da seguinte forma: o remetente grava o tempo em que ele envia um pacote e, se um reconhecimento de sua entrega não retorna dentro de um dado intervalo de tempo, o pacote é retransmitido. Este mecanismo também deverá evitar a aceitação de pacotes duplicados, isto é, o destinatário terá que reconhecer se o pacote recebido é um novo pacote ou um que tenha acabado de ser entregue. Para resolver esse problema, os pacotes são numerados, antes da transmissão, com seqüências de números binários e, para cada pacote recebido, é enviado um reconhecimento carregando a sua seqüência numérica. Na rede-exemplo, a perda de pacotes e a de reconhecimentos são representadas pelas transições **3** e **7**.

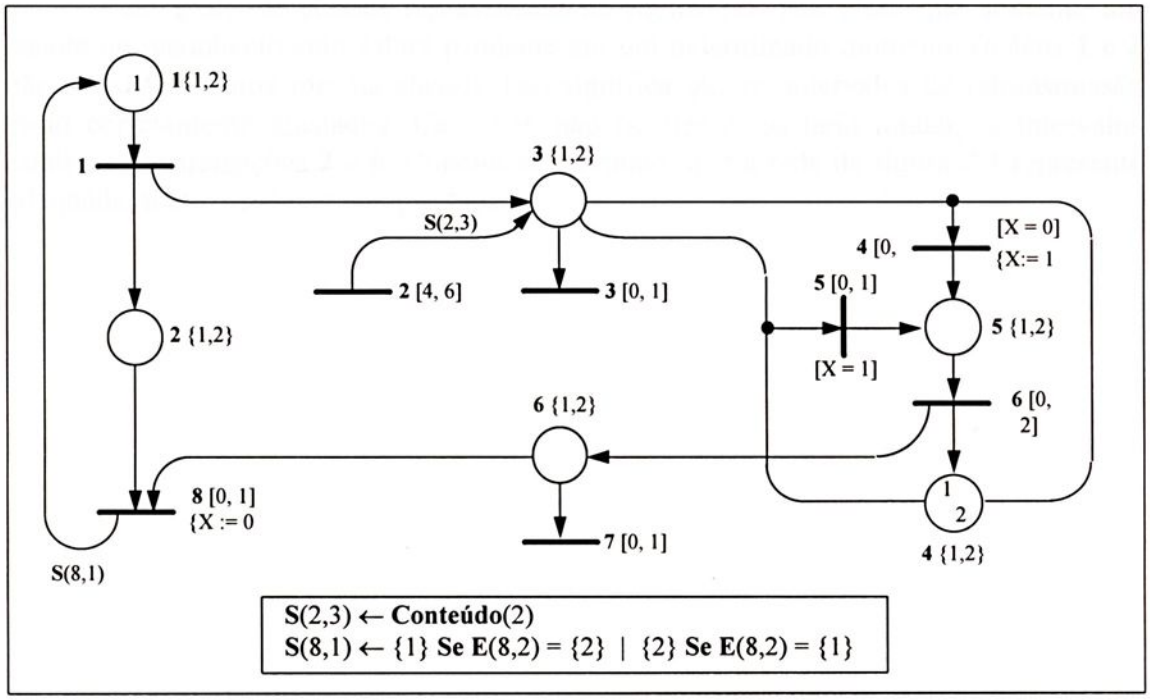


Fig. 7.3. Modelo RPTAN de um Protocolo de Comunicação *Half-Duplex*.

Naturalmente, estimativas para a duração de todas as ações elementares do protocolo devem ser fornecidas. Observa-se que a retransmissão de pacotes (transição **2**) ocorre num tempo compreendido entre 4 e 6 unidades de tempo após o pacote ter sido enviado. Estimativas iguais (entre 0 e 1 unidades de tempo) são dadas para perdas e recepções de pacotes e reconhecimentos. Envio de reconhecimentos têm estimativas entre 0 e 2. Estimativas entre 0 e 0 denotam a necessidade de disparo da transição assim que ficar habilitada (antes do disparo de qualquer outra transição também habilitada).

A figura 7.4 apresenta a correspondência entre cada transição do modelo em RPTAN e a ação correspondente no sistema real. O grafo de classes encontra-se na figura 7.5.

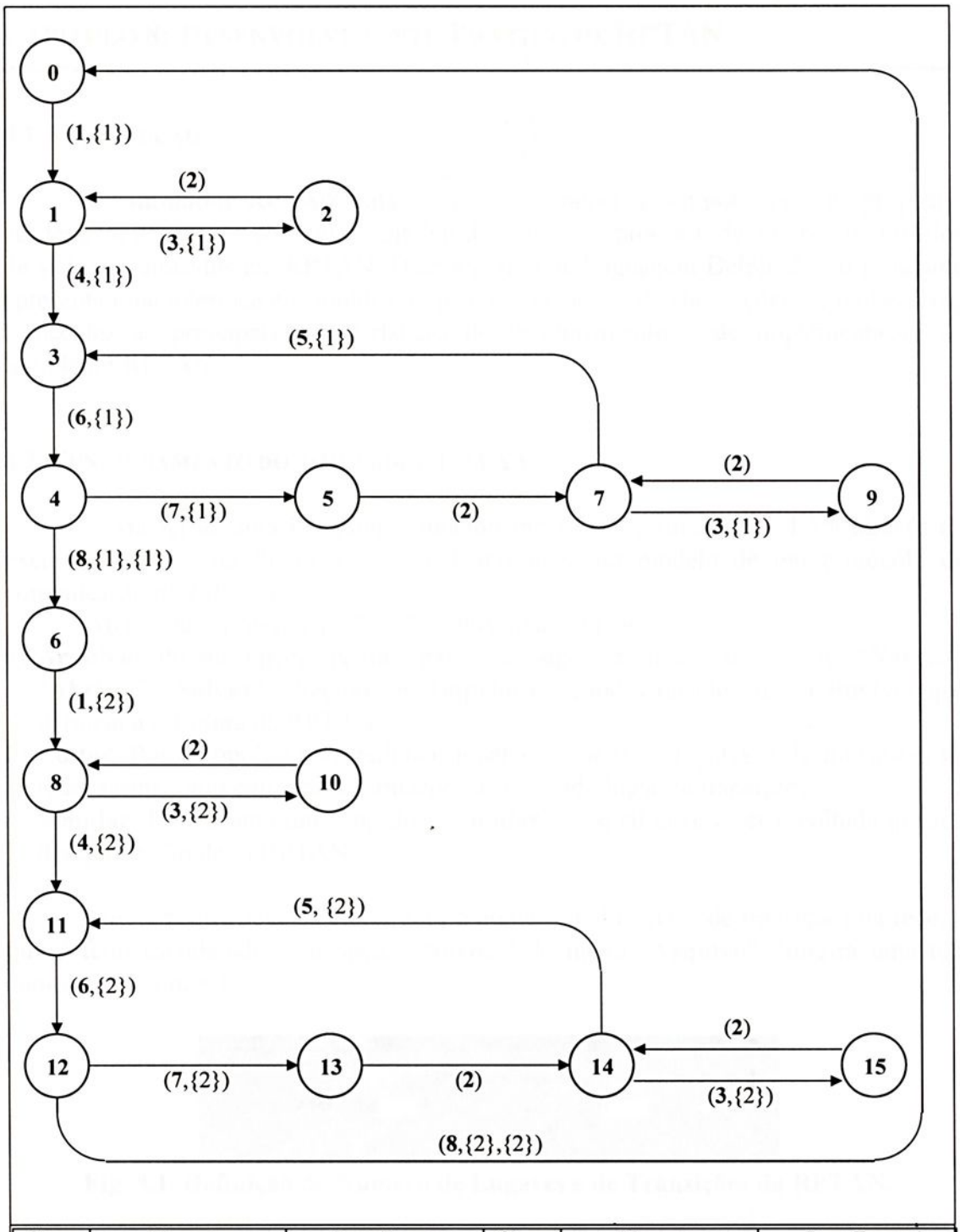
Transição	Ação Correspondente no Sistema
1	Enviar pacote
2	Reenviar pacote
3	Perder pacote
4	Receber e aceitar pacote
5	Receber e rejeitar pacote
6	Enviar reconhecimento de chegada do pacote
7	Perder reconhecimento de chegada do pacote
8	Receber reconhecimento de chegada do pacote

Fig. 7.4. Significado das Transições da Rede da Figura 7.3.

Em relação à análise de propriedades, a rede-exemplo adotada é limitada, reiniciável e viva para a classe inicial adotada (classe 0). O fato de uma RPTAN ser reiniciável implica que o respectivo sistema modelado retorna ao seu estado inicial. Por outro lado, uma RPTAN viva garante um sistema livre de impasses (*deadlocks*).

Pelo grafo de classes representado na figura 7.5, fica claro que somente um pacote ou reconhecimento estará pendente em um determinado momento (*tokens* **1** e **2** não coexistem numa mesma classe). Isto significa que os intervalos de retransmissão estão corretamente ajustados. Caso isto não ocorresse, bastaria mudar os intervalos estáticos das transições **2** e **6**. Conclui-se, portanto, que a rede da figura 7.3 representa adequadamente o sistema em questão.

unesp



Classe	Marcação {(Lugar,Tokens ...), ...}	Dom. Tempo {(Trans, LDI, LDS), ...}	Classe	Marcação {(Lugar,Tokens ...), ...}	Dom. Tempo {(Trans, LDI, LDS), ...}
0	{(1,1),(4,1,2)}	{(1,0,0)}	8	{(2,2),(3,2),(4,1,2)}	{(3,0,1),(4,0,1)}
1	{(2,1),(3,1),(4,1,2)}	{(3,0,1),(4,0,1)}	9	{(2,1),(4,1,2)}	{(2,4,6)}
2	{(2,1),(4,1,2)}	{(2,4,6)}	10	{(2,2),(4,1,2)}	{(2,4,6)}
3	{(2,1),(4,2),(5,1)}	{(2,4,6),(6,0,2)}	11	{(2,2),(4,1),(5,1)}	{(2,4,6),(6,0,2)}
4	{(2,1),(4,1,2),(6,1)}	{(2,2,6),(7,0,1),(8,0,1)}	12	{(2,2),(4,1,2),(6,2)}	{(2,2,6),(7,0,1),(8,0,1)}
5	{(2,1),(4,1,2)}	{(2,1,6)}	13	{(2,2),(4,1,2)}	{(2,1,6)}
6	{(1,2),(4,1,2)}	{(1,0,0)}	14	{(2,2),(3,2),(4,1,2)}	{(3,0,1),(5,0,1)}
7	{(2,1),(3,1),(4,1,2)}	{(3,0,1),(5,0,1)}	15	{(2,2),(4,1,2)}	{(2,4,6)}

Figura 7.5. Grafo de Classes da Rede da Figura 7.3.

CAPÍTULO 8: DESENVOLVIMENTO PRÁTICO DE RPTAN

8.1. INTRODUÇÃO

O **Simulador RPTAN** trata-se de uma ferramenta computacional de propósito acadêmico, desenvolvido com o objetivo de facilitar o processo de análise de modelos de sistemas descritos em RPTAN. Desenvolvido na linguagem Delphi 2.0, o programa apresenta uma interface nos moldes do padrão Windows 95. Nas seções seguintes serão fornecidas as principais características de funcionamento e de implementação do Simulador RPTAN.

8.2. FUNCIONAMENTO DO SIMULADOR RPTAN

Nesta seção, será ilustrado o funcionamento do Simulador RPTAN através do exemplo da rede da figura 7.3, a qual apresenta um modelo de um protocolo de comunicação *half-duplex*.

Ao iniciar o programa, vê-se 3 menus, quais sejam:

a) **Arquivo**: Possui opções-padrão para este tipo de menu, tais como "**Novo...**", "**Abrir...**", "**Salvar**", "**Fechar**" e "**Imprimir...**", todas relacionadas a arquivos que definem a estrutura da RPTAN;
b) **Editar**: Possui opções para redimensionar o número de lugares e de transições da rede, assim como editar as informações de um dado lugar ou transição;
c) **Simular**: Possui uma única opção ("**Simular**"), a qual deverá ser escolhida quando desejar-se simular a RPTAN.

Inicialmente, deve-se entrar com o número de lugares e de transições da rede, o que é feito escolhendo-se a opção "**Novo...**" do menu "**Arquivo**". Surgirá uma tela como a da figura 8.1.

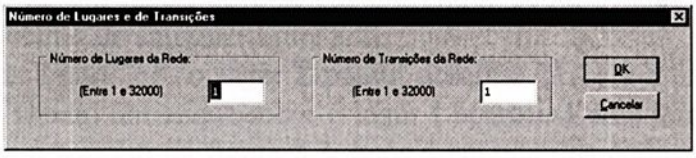


Fig. 8.1. Definição do Número de Lugares e de Transições da RPTAN.

A RPTAN da figura 7.3 possui 6 lugares e 8 transições. Após entrar com estes valores e dar "**OK**", aparecerá a matriz de entrada da rede, conforme a figura 8.2. Caso a rede possuísse mais de 8 lugares ou mais de 8 transições, apareceriam barras de rolagem.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

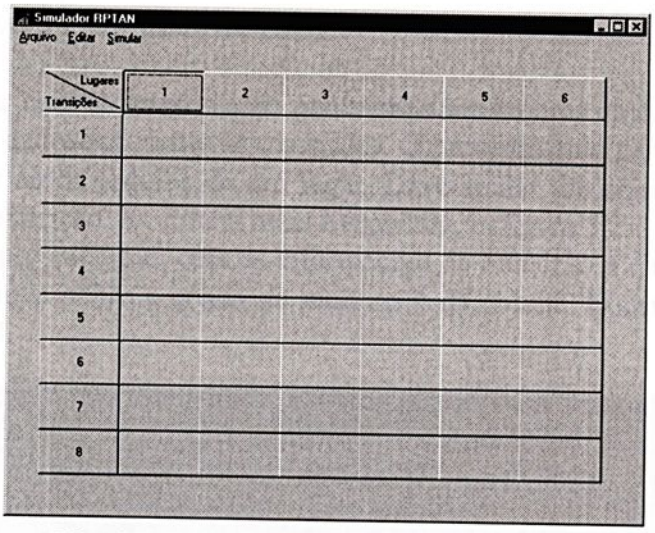


Fig. 8.2. Matriz de Entrada da RPTAN.

A seqüência para a entrada dos dados referentes à estrutura da rede fica a cargo do usuário, da maneira como achar mais conveniente. No texto a seguir, serão definidos inicialmente os **lugares**, seguidos pelas **transições** e, por último, pelos **ramos**. O mais correto é definir os ramos antes das transições, uma vez que estas podem apresentar funções que utilizam ramos como argumentos. Como este não é o caso da RPTAN da figura 7.3, adotou-se a primeira ordem para fins didáticos.

8.2.1. Entrada dos Lugares da Rede

Todo lugar de uma RPTAN possui um **domínio**, o qual designa o conjunto de *tokens* que podem ocupar cada lugar. Além disso, cada lugar pode ser **inicializado** com um subconjunto de *tokens* de seu domínio.

Por *default*, o simulador atribui o conjunto {**0**} para todos os lugares da rede, sendo vazio o conjunto de inicialização de cada lugar. Caso deseje-se alterar o domínio ou a inicialização de um lugar, deve-se dar um clique sobre o número do mesmo, na linha superior da matriz de entrada (figura 8.2).

Na rede da figura 7.3, o lugar **1** possui domínio {**1,2**} e é inicializado com o subconjunto {**1**}. Dando um clique sobre o lugar **1** aparecerá uma janela, a qual deverá apresentar-se com o aspecto da figura 8.3, após a entrada dos dados. Após dar "**OK**", deve-se proceder de forma análoga para os demais lugares da rede.

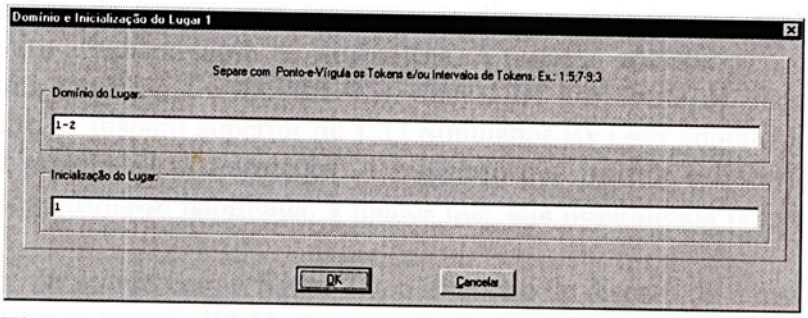


Fig. 8.3. Lugar 1 com Domínio {1,2} e Inicializado com {1}.

8.2.2. ENTRADA DAS TRANSIÇÕES DA REDE

Após entrar com as informações pertinentes aos lugares da rede, pode-se passar para a entrada dos dados referentes às transições. O procedimento para a escolha de uma transição é similar ao da escolha de um lugar. Deve-se dar um clique sobre o número correspondente à transição, na coluna mais à esquerda da figura 8.2.

Entremos, por exemplo, com as informações da transição **4** da RPTAN da figura 7.3. Dando um clique sobre a célula de número 4 da coluna **Transições**, aparecerá a janela da figura 8.4.

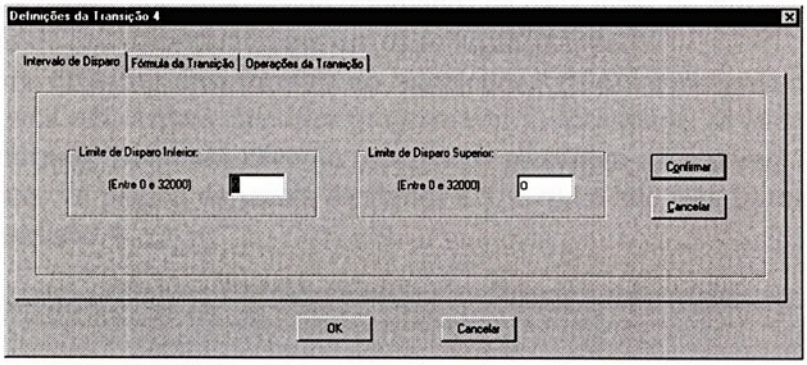


Fig. 8.4. Definições da Transição 4.

A janela da figura 8.4 possui 3 páginas, as quais podem ser selecionadas através das guias **Intervalo de Disparo**, **Fórmula da Transição** e **Operações da Transição**. Cada uma dessas páginas possui os botões "**Confirmar**" e "**Cancelar**" os quais servem, respectivamente, para confirmar ou cancelar os dados presentes na página ativada e passar para a próxima página. Para encerrar a entrada de dados referente à transição, deve-se dar um clique em um dos dois botões localizados na parte inferior da janela **Definições da Transição**, quais sejam, os botões "**OK**" (para aceitar os dados) e "**Cancelar**" (para desprezar quaisquer modificações sobre os dados atuais).

A página correntemente ativa na figura 8.4 é a **Intervalo de Disparo**. Conforme explicado no capítulo 7, o intervalo de disparo de uma transição é um intervalo de reais não-negativos, representando o período de tempo no qual uma transição habilitada pode disparar. Para o Simulador RPTAN, os limites do intervalo de disparo são números inteiros não-negativos.

Na janela da figura 8.4, dá-se a entrada dos limites de disparo **inferior** e **superior** conhecidos por **estáticos**. Em tempo de simulação, os limites estáticos são adotados tão logo a transição seja habilitada. A partir desse ponto, os limites de disparo sofrem alterações, de acordo com as regras de disparo descritas no capítulo anterior, recebendo assim a denominação de **dinâmicos**. A notação adotada para o intervalo de disparo de uma transição **t** é **t[LDI,LDS]**, onde **LDI** é o limite de disparo inferior e **LDS** é o limite de disparo superior de **t**. O Simulador RPTAN adota o intervalo **[0,0]** como *default* para todas as transições. A transição que possuir o intervalo **[0,0]** deve disparar tão logo torna-se habilitada, a menos que seja desabilitada por outra transição concorrente.

Assim, de acordo com a figura 7.3, o intervalo de disparo da transição **4** é **[0,1]**, o que é representado por **4[0,1]**. Ao entrar com estes valores (**0** e **1**) nos campos

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

correspondentes (figura 8.4) e clicar sobre "**Confirmar**", o simulador automaticamente passará para a página **Fórmula da Transição**.

A fórmula de uma transição trata-se de uma expressão booleana calculada em tempo de execução, a qual constitui uma das condições de habilitação da transição correspondente. A fórmula da transição **4** da rede da figura 7.3 é bastante simples:

$$[X = 1].$$

Esta fórmula significa que uma das condições de habilitação da referida transição é que o valor da variável **X** seja igual à **1**. Caso **X** seja diferente de **1**, a transição **4** encontrar-se-á desabilitada. A fórmula da transição poderá conter, ainda, funções que façam referências aos *tokens* que estão sendo removidos de cada lugar de entrada da transição, ou a conjuntos de *tokens* presentes em um dado lugar da rede.

A forma de entrar uma fórmula no Simulador RPTAN é parecida com a entrada de dados em uma calculadora científica programável. Basicamente, deve-se entrar com a fórmula em **notação polonesa reversa** (operandos precedem operadores), usando-se os botões presentes na página da janela da figura 8.5. Repare que esta figura já contém a fórmula da transição **4**.

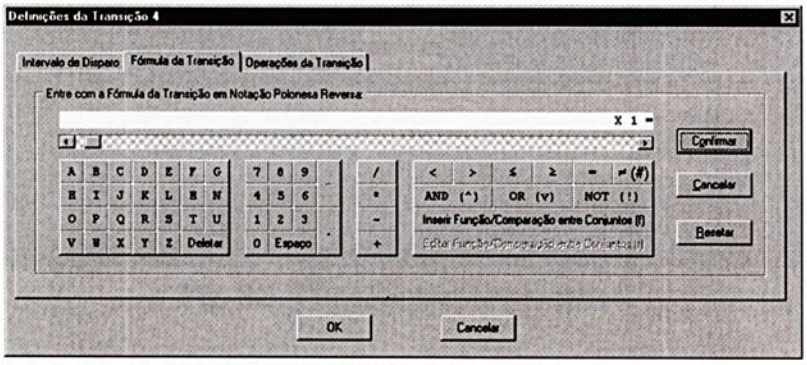


Fig. 8.5. Fórmula da Transição 4.

Acompanhando a figura 8.5, pode-se observar a existência de uma **barra de rolagem**, localizada logo abaixo do **visor** (quadro de texto onde aparece a fórmula). A barra de rolagem serve para avançar ou retroceder o **cursor do visor** (representado por um destaque em azul). O cursor indicará o ponto de **inserção/deleção** de símbolos. Para inserir um símbolo, basta dar um clique sobre o botão correspondente ao símbolo desejado. Por outro lado, para deletar um símbolo, basta destacá-lo com o cursor e, então, clicar sobre o botão "**Deletar**".

Pode-se observar, ainda na figura 8.5, que os botões para entrada de uma fórmula estão divididos em 6 grupos, quais sejam:

a) **Variáveis**: Os botões de "**A**" a "**Z**" representam as variáveis que podem ser utilizadas pelo Simulador RPTAN. As variáveis armazenam números reais;
b) **Números**: Os botões "**.**", "**-**", "**Espaço**" e os numerados de "**0**" a "**9**" permitem a entrada de números reais. Para separar um número de outro, deve-se utilizar o botão "**Espaço**". Já o botão "**-**" deverá ser clicado para inserir o sinal negativo antes de um número;
c) **Operadores Aritméticos**: Os botões "**+**", "**-**", "*****" e "**/**" representam os operadores aritméticos usuais;

d) **Operadores Lógicos**: Os botões "**AND**", "**OR**" e "**NOT**" representam os operadores booleanos de **conjunção**, **disjunção** e **negação**, respectivamente;

e) **Operadores Relacionais**: Os botões "<", ">", "≤", "≥", "=" e "≠" representam os operadores relacionais usuais;

f) **Funções de Comparação Entre Conjuntos**: São funções booleanas entradas e editadas respectivamente pelos botões "**Inserir Função / Comparação entre Conjuntos**" e "**Editar Função / Comparação entre Conjuntos**". No visor, são representadas pela letra "**f**", as quais são diferenciadas pela posição que ocupam no visor. Ao entrar ou editar uma função **f**, surgirá a janela da figura 8.6.

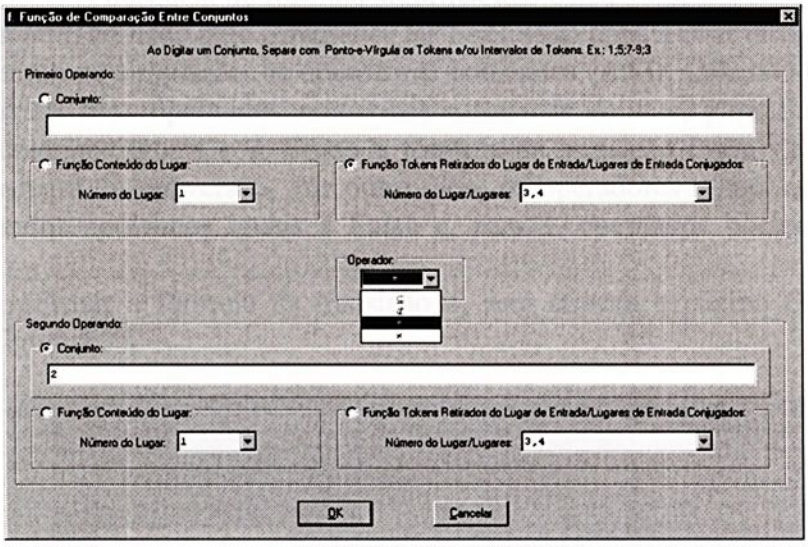


Fig. 8.6. Funções Implementadas.

Observe que a janela apresentada (figura 8.6) é dividida em 3 seções, quais sejam: **Primeiro Operando**, **Operador** e **Segundo Operando**.

Os operandos representam conjuntos, os quais podem ser fornecidos de 3 formas, escolhidas por meio dos seguintes botões de rádio:

f.1) **Conjunto**: Permite a enumeração dos elementos do conjunto, através da caixa de texto localizada logo abaixo do botão de rádio;

f.2) **Função Conteúdo do Lugar**: Esta função retorna o conjunto de *tokens* presentes em dado lugar, no momento de simulação em que é verificada a habilitação ou não da transição (ou seja, na classe corrente). O argumento da função é escolhido dentre os lugares da rede, os quais encontram-se na lista **Número do Lugar**;

f.3) **Função Tokens Retirados do Lugar de Entrada / Lugares de Entrada Conjugados**: Esta função retorna o conjunto de *tokens* removidos por um determinado ramo de entrada, comum ou conjugado, da transição, caso esta venha a disparar. A determinação do argumento da função (ou seja, do ramo) se dá através da escolha dos lugares ligados à transição pelo ramo desejado (lista **Número do Lugar/Lugares**, figura 8.6).

Já o operador poderá ser selecionado através da lista **Operador**, a qual contém os operadores usuais de comparação entre conjuntos ($\subseteq$, $\not\subset$, $=$, $\neq$).

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

No exemplo da figura 8.6, a função **f** retornaria o valor booleano **Verdadeiro** caso o conjunto de *tokens* removido pelo ramo de entrada conjugado que liga os lugares **3** e **4** à transição **4**, fosse igual a {**2**}, quando do disparo da transição **4**. Sendo tal conjunto diferente de {**2**}, a função **f** retornaria o valor booleano **Falso**.

Como a fórmula da transição **4** da rede da figura 7.3 não possui funções de comparação entre conjuntos, deve-se clicar sobre o botão "**Cancelar**" da figura 8.6.

De volta à janela da figura 8.5, observa-se, na parte mais à direita da página **Fórmula da Transição**, os botões "**Confirmar**", "**Cancelar**" e "**Resetar**". Os dois primeiros têm funcionamento trivial. O botão **Resetar** serve para apagar a fórmula presente no visor. A ausência de fórmula significa que o seu valor é **Verdadeiro**, sendo este o valor *default* adotado pelo Simulador RPTAN, logo após a criar uma nova rede.

Em tempo de simulação, os operadores booleanos (**AND, OR, NOT**) convertem para **Verdadeiro** seus operandos numéricos diferentes de **0** e, para **Falso**, os seus operandos numéricos iguais a **0**. Como a fórmula da transição é um valor booleano, adota-se esta regra de conversão. Por outro lado, os operadores numéricos convertem para **0** operandos booleanos iguais a **Falso** e, para **1**, operandos booleanos iguais a **Verdadeiro**.

Ao confirmar a fórmula da transição **4**, será ativada a página **Operações da Transição** (figura 8.7).

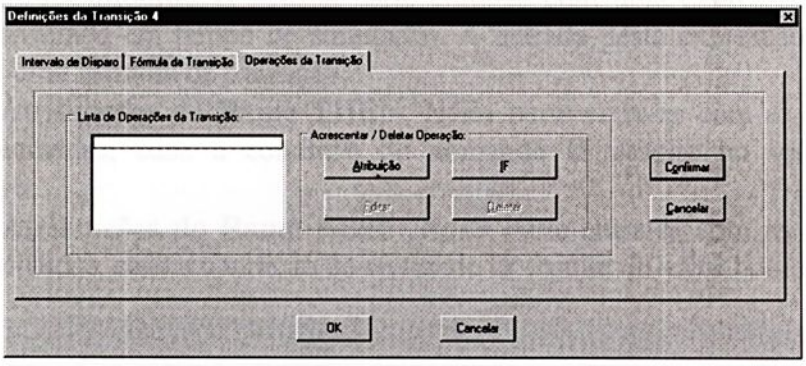


Fig. 8.7. Operações da Transição 4.

Conforme descrito no capítulo anterior, as operações de uma transição podem ser tanto **atribuições** quanto **comandos condicionais** do tipo **IF-THEN-ELSE**.

A transição **4** da rede da figura 7.3 possui uma única operação, a qual atribui à variável **X** o valor **1**. Clicando sobre o botão "**Atribuição**" da figura 8.7, aparecerá a janela para entrada da atribuição. Tal janela deverá apresentar o aspecto da figura 8.8, logo após a entrada da atribuição.

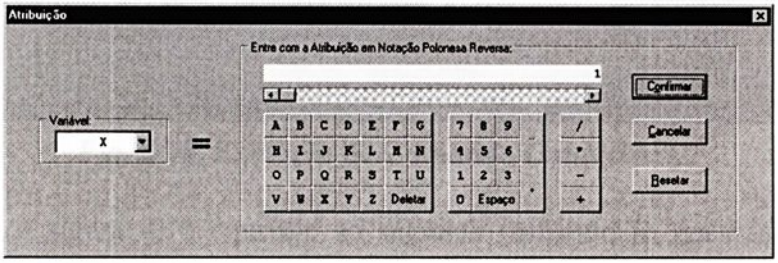


Fig. 8.8. Entrada da Atribuição da Transição 4.

Os botões da janela da figura 8.8 possuem funcionamento semelhante ao da página **Fórmula da Transição** (figura 8.5).

Após confirmar a atribuição da transição **4**, a lista de operações da mesma será atualizada, conforme mostrado na figura 8.9.

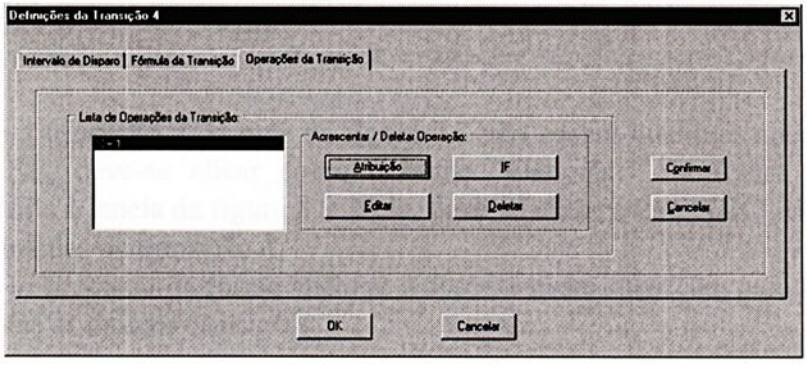


Fig. 8.9. Atualização da Lista de Operações da Transição 4.

Para entrar com um comando do tipo IF-THEN-ELSE, deve-se clicar sobre o botão "**IF**" (figura 8.9). Surgirá, então, uma janela dividida em 3 seções, quais sejam (observe a figura 8.10):

a) **Condição do Comando IF**: Nesta parte entra-se com uma expressão booleana, a qual será avaliada em tempo de simulação. A entrada desta expressão é idêntica à entrada de uma fórmula de transição;

b) **Lista de Atribuições do Ramo THEN**: Nesta parte entra-se com as atribuições a serem executadas, caso a condição do comando IF resulte no valor booleano **Verdadeiro**;

c) **Lista de Atribuições do Ramo ELSE**: Nesta parte entra-se com as atribuições a serem executadas, caso a condição do comando IF resulte no valor booleano **Falso**;

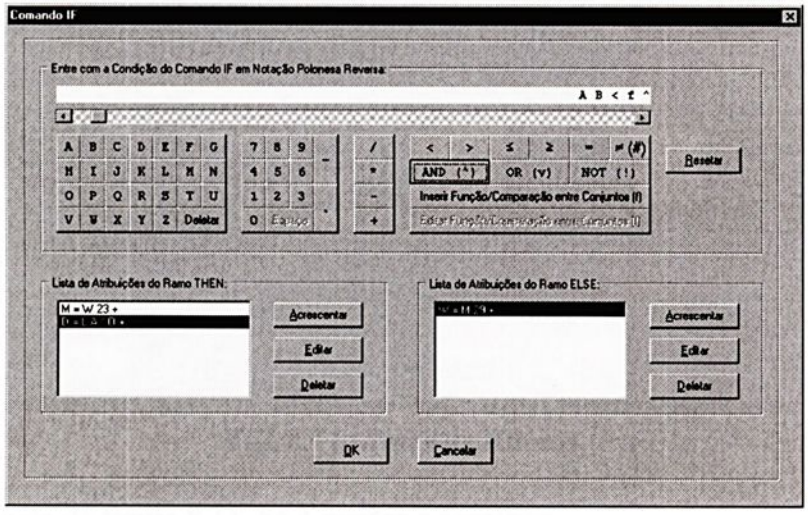


Fig. 8.10. Exemplo de Comando Condicional.

Assim, a figura 8.10 equivale ao seguinte comando IF em Pascal (considere definida alguma função **f( )**):

```
if  ((A<B) and f( ))
then
    begin
        M := W + 23;
        D := (L*A) + O;
    end
else
    W := M + 29;
```

Como a transição **4** da rede da figura 7.3 não possui qualquer operação do tipo IF-THEN-ELSE, deve-se clicar sobre o botão "**Cancelar**" da figura 8.10, o que remeterá de volta à janela da figura 8.9. Nela, deve-se clicar sobre "**OK**", encerrando-se, assim, as definições da transição **4**.

Deve-se proceder de forma análoga à descrita nesta subseção, para a entrada dos dados referentes às demais transições.

8.2.3. ENTRADA DOS RAMOS DA REDE

RPTAN comportam 3 tipos de ramos, os quais foram descritos no capítulo 7. Para entrar com as definições de um dado ramo, deve-se clicar sobre a célula correspondente ao cruzamento entre a transição e o lugar (ou um dos lugares, em se tratando de ramo de entrada conjugado). Por exemplo, clicando sobre a célula presente no cruzamento entre a primeira linha e primeira coluna da matriz de entrada (figura 8.2), surgirá a janela da figura 8.11.

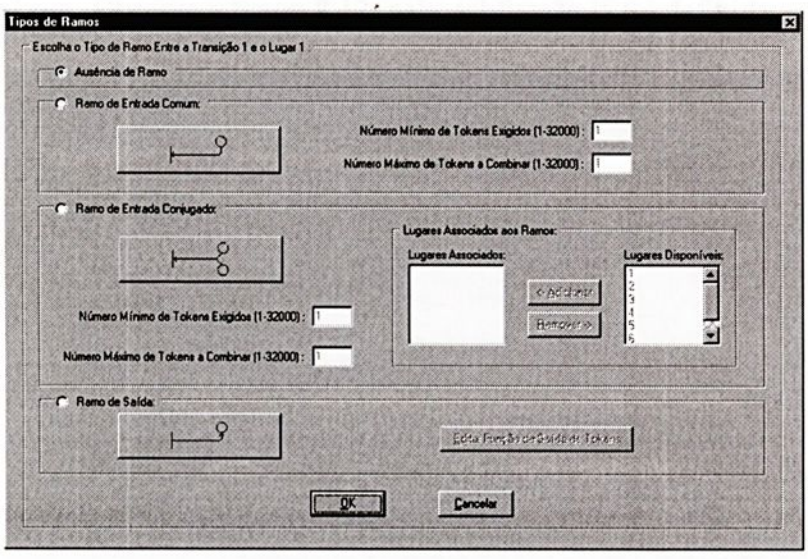


Fig. 8.11. Tipos de Ramos.

Na janela da figura 8.11, há 4 opções a escolher:

a) **Ausência de Ramo**: Opção usada para limpar a célula;
b) **Ramo de Entrada Comum**: Opção usada para denotar um ramo de entrada comum, o qual removerá do lugar correspondente, quando do disparo da transição, uma combinação de *tokens* de cardinalidade entre **Número Mínimo de Tokens Exigidos**

e **Número Máximo de Tokens a Combinar**. O Simulador RPTAN adota, por *default*, combinações de tamanho 1;

c) **Ramo de Entrada Conjugado**: Opção usada para denotar um ramo de entrada conjugado, o qual possui comportamento análogo ao de um ramo de entrada comum, com a diferença de atuar simultaneamente sobre um conjunto de lugares, sendo tais lugares denominados **conjugados**;

d) **Ramo de Saída**: Opção usada para denotar um ramo de saída, o qual envia um conjunto de *tokens* para o lugar correspondente, quando do disparo da transição. O conjunto enviado será explicado mais adiante.

Para selecionar o opção desejada, deve-se clicar sobre o botão de rádio correspondente ou sobre o botão com o desenho do ramo. No caso da RPTAN da figura 7.3, existe um ramo de entrada comum entre o lugar **1** e a transição **1**. Assim, deve-se clicar o botão de rádio "**Ramo de Entrada Comum**". Como o ramo da figura 7.3 não tem qualquer referência ao **Número Mínimo de Tokens Exigidos** e **Número Máximo de Tokens a Combinar**, assume-se para os mesmos o valor 1. Clicando "**OK**" na janela da figura 8.11, a matriz de entrada da figura 8.2 será atualizada. Os demais ramos de entrada poderão ser entrados sem maiores dificuldades.

Agora, passa-se à definição de um ramo de saída com função. Este é o caso do ramo ligando a transição **2** ao lugar **3**, o qual repassa para o lugar **3** o conjunto de *tokens* presentes no lugar **2**, quando do disparo da transição **2**. Para definir tal ramo, deve-se clicar sobre a célula (**2**,**3**) da matriz de entrada. Em seguida, deve-se clicar sobre o botão de rádio "**Ramo de Saída**", o que tornará habilitado o botão "**Editar Função de Saída de Tokens**" (desabilitado na figura 8.11, uma vez que a opção em foco era **Ausência de Ramo**). Clicando sobre este último botão, aparecerá a janela para a definição da função de saída de *tokens* . Observe a figura 8.12; nela, tal função já foi definida.

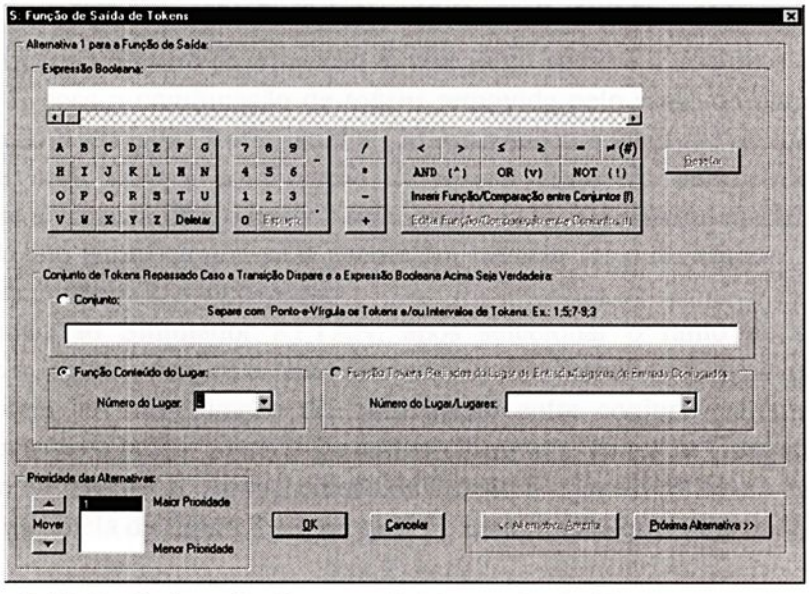


Fig. 8.12. Definição da Função do Ramo de Saída da Transição 2.

A função de saída de *tokens* poderá ser formada por várias **alternativas**. Cada alternativa é composta for uma **expressão booleana** e um **conjunto**. As alternativas seguem uma **ordem de prioridade** de verificação de sua expressão booleana. A

primeira das alternativas a atender as condições de habilitação da transição (conforme descrito no capítulo 7), repassará o conjunto de *tokens* correspondente para o lugar de saída.

A expressão booleana da alternativa é entrada da mesma forma como a fórmula de uma transição, enquanto os conjuntos são definidos da forma descrita anteriormente, a qual é transcrita a seguir:

a) **Conjunto**: Permite a enumeração dos elementos do conjunto, através da caixa de texto localizada logo abaixo do botão de rádio;
b) **Função Conteúdo do Lugar**: Esta função retorna o conjunto de *tokens* presentes em dado lugar, no momento de simulação em que é verificada a habilitação ou não da transição (ou seja, na classe corrente). O argumento da função é escolhido dentre os lugares da rede, os quais encontram-se na lista **Número do Lugar**;
c) **Função Tokens Retirados do Lugar de Entrada / Lugares de Entrada Conjugados**: Esta função retorna o conjunto de *tokens* removidos por um determinado ramo de entrada, comum ou conjugado, da transição, caso esta venha a disparar. A determinação do argumento da função (ou seja, do ramo) se dá através da escolha dos lugares ligados à transição pelo ramo desejado (lista **Número do Lugar/Lugares**, figura 8.12).

Conforme dito anteriormente, a ausência da expressão booleana significa que a mesma assume o valor **Verdadeiro**. Assim, pela figura 8.12, vê-se que o disparo da transição **2** promoverá a passagem de uma cópia do conjunto de *tokens* presente no lugar **2** para o lugar **3**, conforme desejado.

Após a confirmação dos dados do ramo de saída entre a transição **2** e o lugar **3**, segue-se a definição dos dados do ramo de saída entre a transição **8** e o lugar **1**. Este ramo ilustra as funções de saída com mais de uma alternativa. Sua função de saída é dada por:

$$\mathbf{S(8,1) \leftarrow \{1\}\ Se\ E(8,2) = \{2\} \mid \{2\}\ Se\ E(8,2) = \{1\}}$$

significando que, ao disparo da transição **8**, este ramo passará o conjunto de *tokens* {**1**} para o lugar **1**, caso a combinação de *tokens* removida pelo ramo de entrada existente entre a transição **8** e o lugar **2** seja igual a {**2**}, ou passará o conjunto de *tokens* {**2**} caso contrário (ou seja, caso a combinação de *tokens* removida pelo ramo de entrada existente entre a transição **8** e o lugar **2** seja igual a {**1**}). No sistema modelado pela rede da figura 7.3, isto significa que se o pacote em trânsito foi o de número **1**, o próximo deverá ser o de número **2**, e vice-versa.

Voltando ao Simulador RPTAN, após selecionar o ramo de saída entre a transição **8** e o lugar **1**, surgirá a tela para entrada da alternativa **1**, na qual a expressão booleana deverá ser uma função de comparação entre conjuntos **f**, que deverá ser entrada clicando-se o botão "**Inserir Função / Comparação entre Conjuntos**". Surgirá então a janela da figura 8.6. Após a edição da função **f**, a janela da figura 8.6 apresentará a disposição da janela da figura 8.13, na qual **f** é uma função booleana dada por:

$$\mathbf{E(8,2) = \{2\}}$$

ou seja, **f** assumirá o valor booleano **Verdadeiro**, caso a combinação de *tokens* retirada do lugar **2** pelo disparo da transição **8** seja igual a {**2**}, e assumirá o valor booleano **Falso**, caso contrário.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

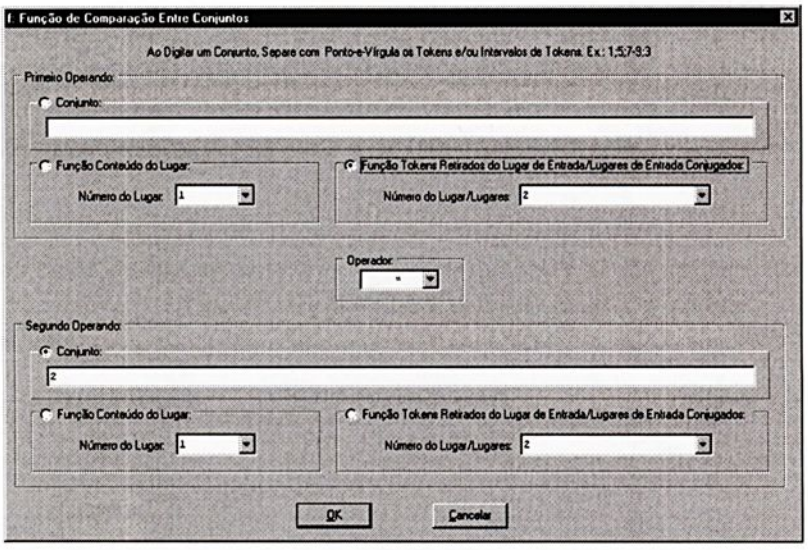


Fig. 8.13. Definição da Função de Comparação Entre Conjuntos da Primeira Alternativa da Função do Ramo de Saída da Transição 8.

Após clicar "**OK**" na janela da figura 8.13, aparecerá a janela da figura 8.14, na qual a expressão booleana da alternativa **1** é composta pela função **f** descrita no parágrafo anterior. Observe que na janela da figura 8.14 já foi definido o conjunto a ser repassado para o lugar **1**, caso **f** seja verdadeira e a transição **8** dispare, qual seja o conjunto {**1**}.

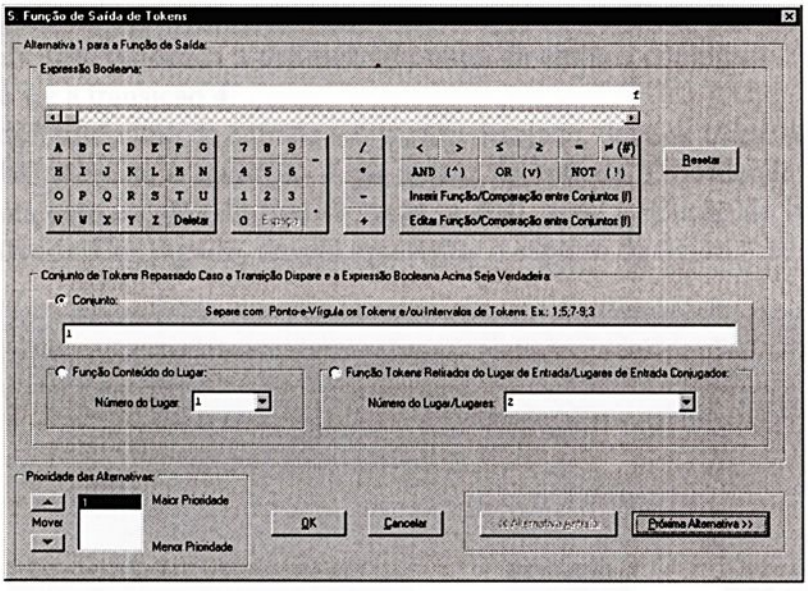


Fig. 8.14. Definição da Primeira Alternativa da Função do Ramo de Saída da Transição 8.

Definida a alternativa **1**, para passar para a próxima alternativa a ser definida, deve-se clicar sobre o botão "**Próxima Alternativa**", presente na janela da figura 8.14. Assim, surgirá a janela da figura 8.15 (já editada), onde a expressão booleana assume sempre o valor **Verdadeiro**, e na qual o conjunto repassado para o lugar **1** com o disparo da transição **8** é dado por {**2**}.

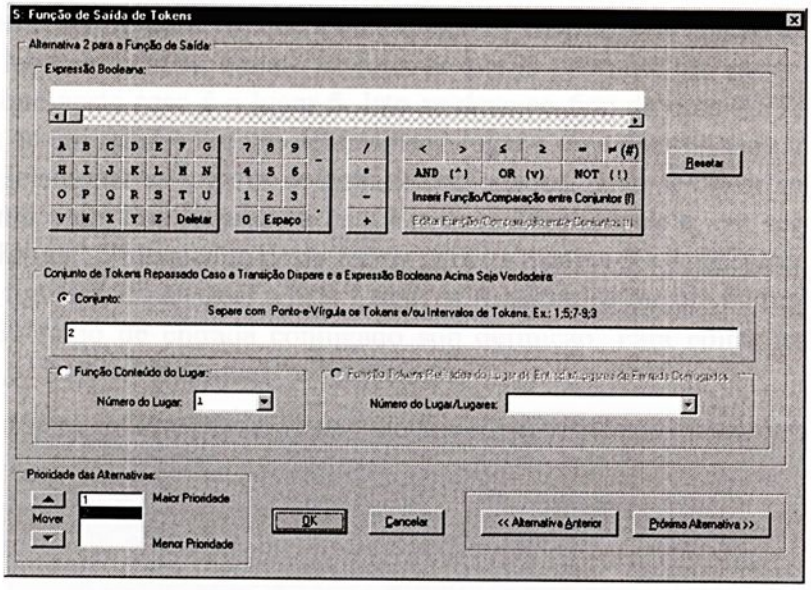


Fig. 8.15. Definição da Segunda Alternativa da Função do Ramo de Saída da Transição 8.

Destarte, após clicar em "**OK**", têm-se definidas as duas alternativas da função de saída de *tokens* presentes no ramo de saída existente entre a transição **8** e o lugar **1** da RPTAN da figura 7.3. Os demais ramos de saída poderão ser definidos sem maiores dificuldades.

Finalmente, passa-se à definição dos ramos de entrada conjugados. O exemplo da rede da figura 7.3 apresenta dois ramos desse tipo, os quais são ligados às transições **4** e **5**. A seguir, será mostrada a definição do ramo de entrada conjugado existente entre os lugares **3** e **4** e a transição **4**.

Para definir o ramo de entrada conjugado desejado, deve-se clicar sobre qualquer célula da linha referente à transição **4**, na matriz de entrada. Procedendo dessa forma, surgirá a janela para escolha do tipo de ramo, a qual, após a seleção do botão de rádio **Ramo de Entrada Conjugado**, deverá apresentar o aspecto da figura 8.16.

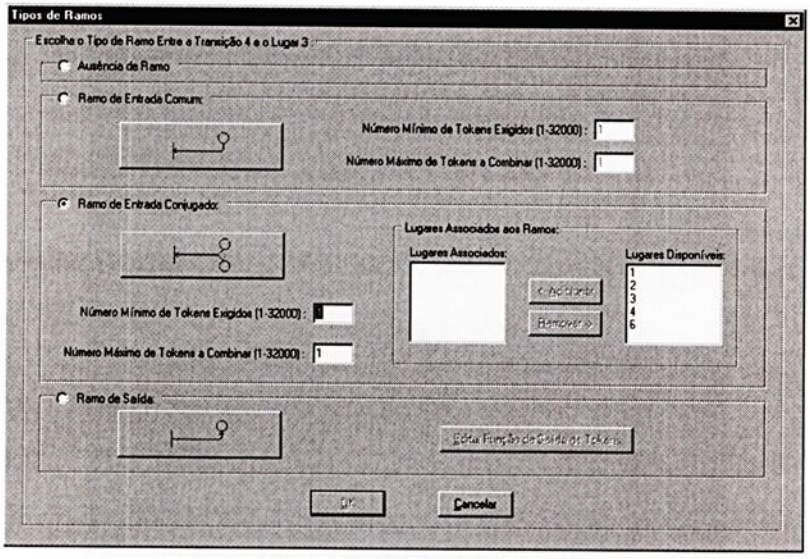


Fig. 8.16. Esperando a Definição do Ramo de Entrada Conjugado da Transição 4.

Observe, na figura 8.16, que a lista **Lugares Disponíveis** apresenta a numeração de todos os lugares da rede, exceto os lugares que já possuam um ramo relacionado à transição **4**, que neste caso é o lugar **5**. Isto se deve ao fato de uma RPTAN não poder possuir mais de um ramo entre um determinado lugar e uma determinada transição.

Para selecionar os lugares de entrada conjugados, deve-se clicar sobre o número correspondente a um dos lugares na lista **Lugares Disponíveis** e, em seguida, sobre o botão "**Adicionar**". Procedendo desta forma para os lugares **3** e **4**, chega-se à disposição mostrada na figura 8.17, na qual a lista **Lugares Associados** mostra os lugares ligados à transição pelo ramo de entrada conjugado sob definição. Para editar ambas as listas deve-se usar os botões "**Remover**" e "**Adicionar**".

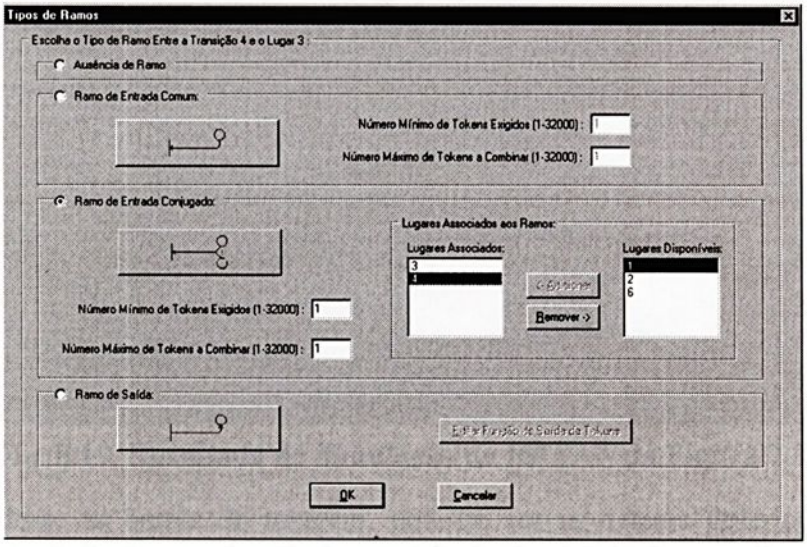


Fig. 8.17. Definição do Ramo de Entrada Conjugado da Transição 4.

Clicando "**OK**", a matriz de entrada é atualizada (figura 8.18). Por fim, deve-se ressaltar que um ramo de entrada conjugado necessita de dois lugares, no mínimo. Dessa forma, caso houve-se a definição de outro tipo de ramo entre a transição **4** e um de seus lugares de entrada conjugados (**3** ou **4**), o ramo de entrada conjugado seria removido.

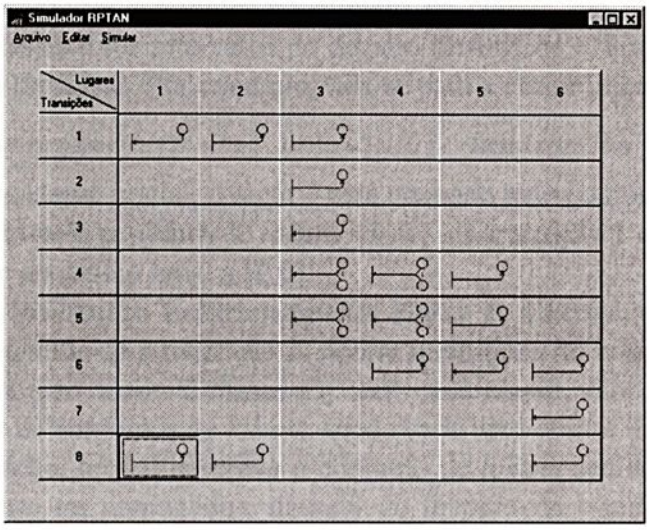


Fig. 8.18. Matriz de Entrada Após Definidos os Ramos da Rede da Figura 7.3.

8.2.4. SIMULAÇÃO DA REDE

Após definida a estrutura da rede, para fazer a sua simulação, deve-se escolher a opção "**Simular**" do menu de mesmo nome.

Terminada a simulação, é apresentada a janela da figura 8.19.

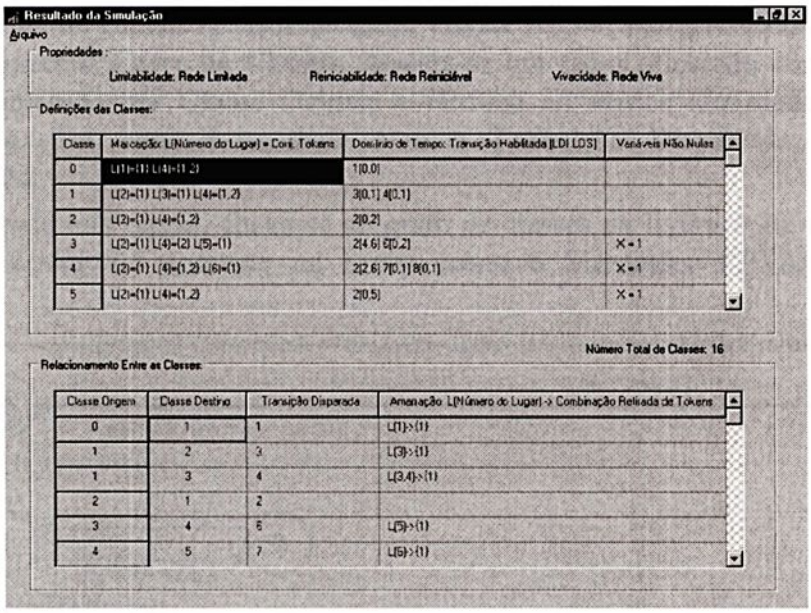


Fig. 8.19. Resultado de Simulação da RPTAN da Figura 7.3.

Usando-se as barras de rolagem, pode-se ver as demais descrições. Para um manuseio mais fácil, deve-se imprimir o resultado de simulação, escolhendo-se o item "**Imprimir**" do menu **Arquivo**.

Na parte superior da janela da figura 8.19, encontra-se a validação das propriedades de **limitabilidade**, **reiniciabilidade** e **vivacidade**. Caso existissem transições não disparadas na simulação, as mesmas seriam listadas junto no quadro **Propriedades**.

Continuando a descrição da figura 8.19, no quadro **Definições das Classes**, encontram-se as definições referentes ao número da classe, sua marcação, transições habilitadas e seus respectivos intervalos de disparo dinâmicos e, ainda, a valoração das variáveis não-nulas da classe. Por exemplo, tem-se para a classe inicial ($\mathbf{C_0}$)

```
Classe: 0  Marcação: L(1)={1} L(4)={1,2}  Domínio de Tempo: 1[0,0]
```

significando que, na classe inicial, o lugar **1** está ocupado pelo conjunto de *tokens* {**1**}, o lugar **4** está ocupado pelo conjunto de *tokens* {**1**, **2**} e a transição **1** está habilitada, com intervalo de disparo dinâmico igual a [**0**,**0**].

Verifica-se, no quadro **Relacionamento Entre as Classes**, que, estando a rede na classe $\mathbf{C_0}$ e disparando-se a transição **1**, ocorre a remoção do conjunto de *tokens* {**1**} do lugar **1**, chegando-se assim à classe $\mathbf{C_1}$. Em cada classe, são calculadas todas as possíveis remoções de conjuntos de *tokens*, sendo cada uma dessas remoções designadas pelo termo **amarração**, o qual descreve o conjunto de *tokens* cedido por cada lugar de entrada (seja comum ou conjugado), quando do disparo da transição. Sendo nula a amarração, significa que a transição não possui ramos de entrada. Tal transição recebe o

nome de **transição somente geradora**. Este é o caso da transição **2**. Observe, na figura 8.19, que a amarração é nula quando a transição disparada for a **2**.

Com base nas informações do simulador, constrói-se o grafo de classes da RPTAN da seguinte forma:

a) Para cada classe do quadro **Definições das Classes**, associa-se um vértice do grafo numerado pelo número da classe;
b) Para cada par de classes (**Classe Origem**, **Classe Alcançada**) pertencente ao quadro **Relacionamento Entre as Classes**, associa-se um ramo orientado com origem no vértice numerado por **Classe Origem** e término no vértice numerado por **Classe Alcançada**. Tal ramo é rotulado com a **Transição Disparada**, seguida da **Amarração**.

Seguindo tais regras, chega-se ao grafo de classes da figura 8.20, o qual difere daquele da figura 7.5 somente na forma como o Simulador RPTAN denota as amarrações.

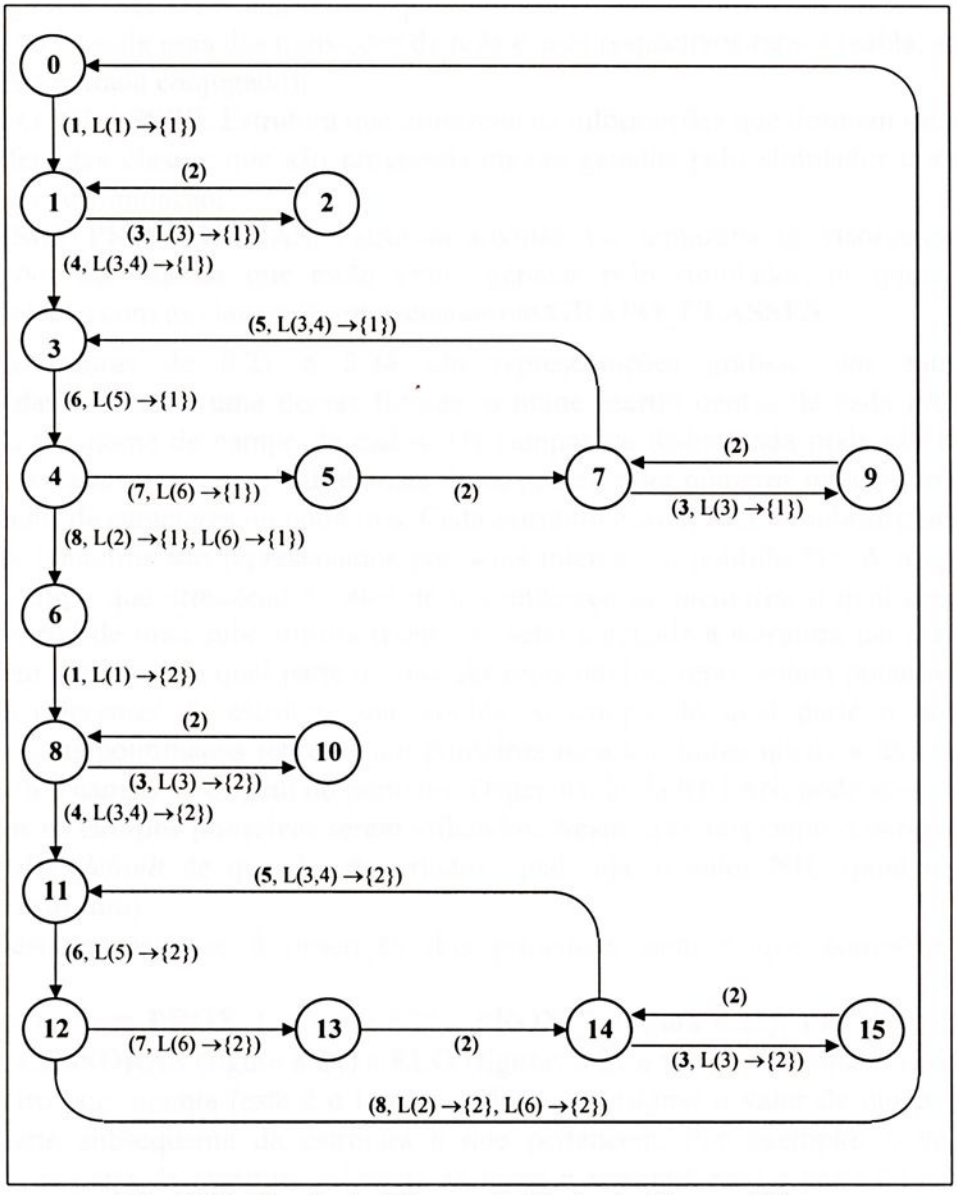


Fig. 8.20. Grafo de Classes da Rede da Figura 7.3.

8.3. Características de Implementação do Simulador RPTAN

8.3.1. Estruturas de Dados

Nesta subseção, serão descritas as estruturas de dados e, nas próximas, os algoritmos das principais partes do Simulador RPTAN.

Em relação às estruturas de dados, deve-se ressaltar que as mesmas são dinâmicas, ou seja, foram implementadas usando-se ponteiros. Desta forma, o tamanho que tais estruturas podem alcançar fica limitado à capacidade de gerenciamento de memória do compilador *Borland Delphi 2.0 Developer For Windows*, no qual o Simulador RPTAN foi desenvolvido.

São 4 as principais estruturas de dados:

a) **INFO_LUGARES**: Estrutura que armazena as informações pertinentes à definição de cada um dos lugares da rede;
b) **INFO_TRANSICOES**: Estrutura que armazena as informações referentes à definição de cada uma das transições da rede e seus respectivos ramos (saída, entrada comum e entrada conjugado);
c) **GRAFO_CLASSES**: Estrutura que armazena as informações que definem cada uma das diferentes classes que são progressivamente geradas pelo simulador durante o processo de simulação;
d) **CLASSES_PROVISORIAS**: Estrutura auxiliar que armazena provisoriamente as definições das classes que estão sendo geradas pelo simulador, as quais serão confrontadas com as classes já armazenadas em **GRAFO_CLASSES**.

As figuras de 8.21 a 8.24 são representações gráficas das estruturas mencionadas. Em cada uma dessas figuras, o nome escrito dentro de cada retângulo representa um nome de campo de dados. Os campos de dados nada mais são do que variáveis, as quais pertencem a diferentes tipos, podendo ser números inteiros, números reais, cadeias de caracteres ou ponteiros. Cada estrutura é composta de subestruturas.

Os ponteiros são representados por setas inteiras ou pontilhadas. A origem da seta é o campo que armazena o valor de um endereço de memória, o qual designa a posição inicial de uma subestrutura (ponta da seta) agregada à estrutura pai (estrutura que contém o campo do qual parte a seta). As setas inteiras representam ponteiros para estruturas diferentes da estrutura que contém o campo do qual parte o ponteiro, enquanto setas pontilhadas representam ponteiros para estruturas iguais à da estrutura que contém o campo de origem do ponteiro. Dependendo da RPTAN, pode acontecer de **nem todos os campos ponteiros** serem utilizados. Neste caso, tais campos permanecem com o valor *default* de quando são criados, qual seja, o valor **NIL** (ponteiro para nenhuma estrutura).

Destarte, passa-se à descrição dos principais campos que compõem cada estrutura.

Os campos **PROX_L** (figura 8.21), **PROX_T** (figura 8.22), **PROX_C** (figura 8.23), **SUCESSORAS** (figura 8.24) e **ELO** (figuras 8.21 a 8.24) armazenam, cada qual, um ponteiro que aponta (este é o termo usado para designar o valor de um ponteiro) para a parte subsequente da estrutura a que pertencem. Por exemplo, o valor de **PROX_L** da parte da estrutura referente ao lugar **n** apontará para a parte da estrutura

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

referente ao lugar **n+1**, caso **n** não seja o último lugar da rede. Sendo **n** o número do último lugar, **PROX_L** armazenará **NIL**.

Os principais campos da estrutura **INFO_LUGARES** (figura 8.21) são:

a) **LUGAR**: Armazena um número inteiro positivo, representando o número do lugar;

b) **DOM_LUGAR**: Armazena uma *AnsiString* (cadeia de caracteres dinâmica que pode chegar até 2 Gb de memória) que guarda, de forma compactada, o conjunto domínio do lugar, ou seja, o conjunto de *tokens* que podem ocupar o lugar;

c) **INICIALIZACAO**: Armazena uma *AnsiString* que guarda, de forma compactada, o conjunto de *tokens* presentes no lugar quando da inicialização da rede;

d) **TOKENS_ATUAIS**: Campo usado durante o processo de simulação. Armazena um ponteiro para a lista de *tokens* que ocupam o lugar, na classe que está sendo simulada. Esta lista é formada por células (estruturas) de dois campos, quais sejam:
   d.1) **TOKEN**: Armazena um número inteiro positivo, representante de um *token*;
   d.2) **ELO**: Conforme explicado anteriormente, aponta para a célula subsequente, qual seja, a estrutura representativa do próximo *token* da lista, se houver;

e) **TOKENS_FUTUROS**: Campo usado durante o processo de simulação. Armazena um ponteiro para a lista de *tokens* que ocupam o lugar, em uma classe que está sendo gerada a partir da classe que está sendo simulada. Esta lista é formada por estruturas de dois campos, iguais à da lista **TOKENS_ATUAIS**.

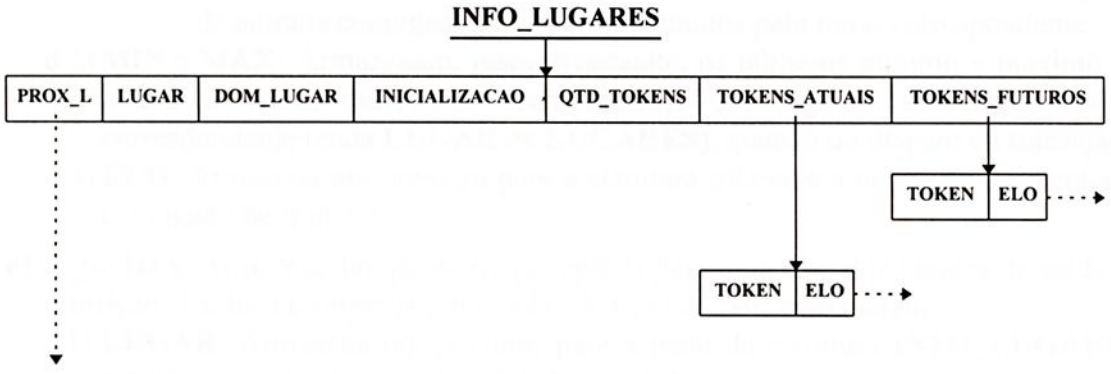


Fig. 8.21. Estrutura de Dados *INFO_LUGARES*.

Em relação à estrutura **INFO_TRANSICOES** (figura 8.22), seus principais campos são:

a) **TRANSICAO**: Armazena um número inteiro positivo, representando o número da transição;

b) **POTENCIALMENTE_DISPARAVEL**: Armazena o valor booleano **TRUE** caso a transição seja disparada pelo menos uma vez durante a simulação, caso contrário, armazena o valor booleano **FALSE**. O valor deste campo é utilizado pelo algoritmo de verificação da propriedade de vivacidade, para identificar as transições que nunca disparam;

c) **L_COMUNS**: Armazena um ponteiro que aponta para uma lista dos lugares de entrada comuns da transição. Tal lista é formada por células de quatro campos, quais sejam:
   c.1) **LUGAR**: Armazena um ponteiro para a parte da estrutura **INFO_LUGARES** relativa a um dos lugares de entrada comuns da transição;
   c.2) **MIN** e **MAX**: Armazenam, respectivamente, os números mínimo e máximo de *tokens* a serem removidos do lugar apontado por **LUGAR** quando do disparo da transição;
   c.3) **ELO**: Armazena um ponteiro para a célula referente a outro dos lugares de entrada comuns da transição;

d) **L_CONJUGADOS**: Armazena um ponteiro que aponta para uma lista dos ramos de entrada conjugados da transição. Por sua vez, cada elemento dessa lista aponta para uma lista dos lugares de entrada conjugados da transição unidos pelo ramo correspondente. A lista dos ramos de entrada conjugados é formada por células de quatro campos, quais sejam:
   d.1) **LUGARES**: Armazena o ponteiro que aponta para a lista dos lugares de entrada conjugados da transição unidos pelo ramo correspondente. Esta lista deve possuir, no mínimo, duas células formadas pelos campos:
      d.1.1) **LUGAR**: Armazena um ponteiro para a parte da estrutura **INFO_LUGARES** relativa a um dos lugares de entrada conjugados da transição, unidos à mesma pelo ramo correspondente;
      d.1.2) **ELO**: Armazena um ponteiro para a célula referente a outro dos lugares de entrada conjugados da transição, unidos pelo ramo correspondente;
   d.2) **MIN** e **MAX**: Armazenam, respectivamente, os números mínimo e máximo de *tokens* a serem removidos conjuntamente dos lugares unidos pelo ramo correspondente (cada **LUGAR** de **LUGARES**), quando do disparo da transição;
   d.3) **ELO**: Armazena um ponteiro para a estrutura referente a outro ramo de entrada conjugado da transição;

e) **L_SAIDA**: Armazena um ponteiro que aponta para uma lista dos lugares de saída da transição. Tal lista é formada por células de três campos, quais sejam:
   e.1) **LUGAR**: Armazena um ponteiro para a parte da estrutura **INFO_LUGARES** relativa a um dos lugares de saída da transição;
   e.2) **FUNCAO**: Armazena um ponteiro que aponta para a estrutura representativa da função de saída de *tokens*, a qual pode possuir diversas alternativas (ver capítulo anterior);
   e.3) **ELO**: Armazena um ponteiro para a célula referente a outro dos lugares de saída da transição;

f) **FORMULA**: Armazena uma *AnsiString*, representante da fórmula da transição;

g) **LDI_E**, **LDS_E**: Armazenam, respectivamente, os limites de disparo inferior e superior estáticos da transição;

h) **LDI_D**, **LDS_D**: Armazenam, respectivamente, os limites de disparo inferior e superior dinâmicos da transição, durante o processo de simulação;

i) **OPERACOES**: Armazena um ponteiro que aponta para a estrutura representativa do conjunto de operações a serem realizadas quando do disparo da transição.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

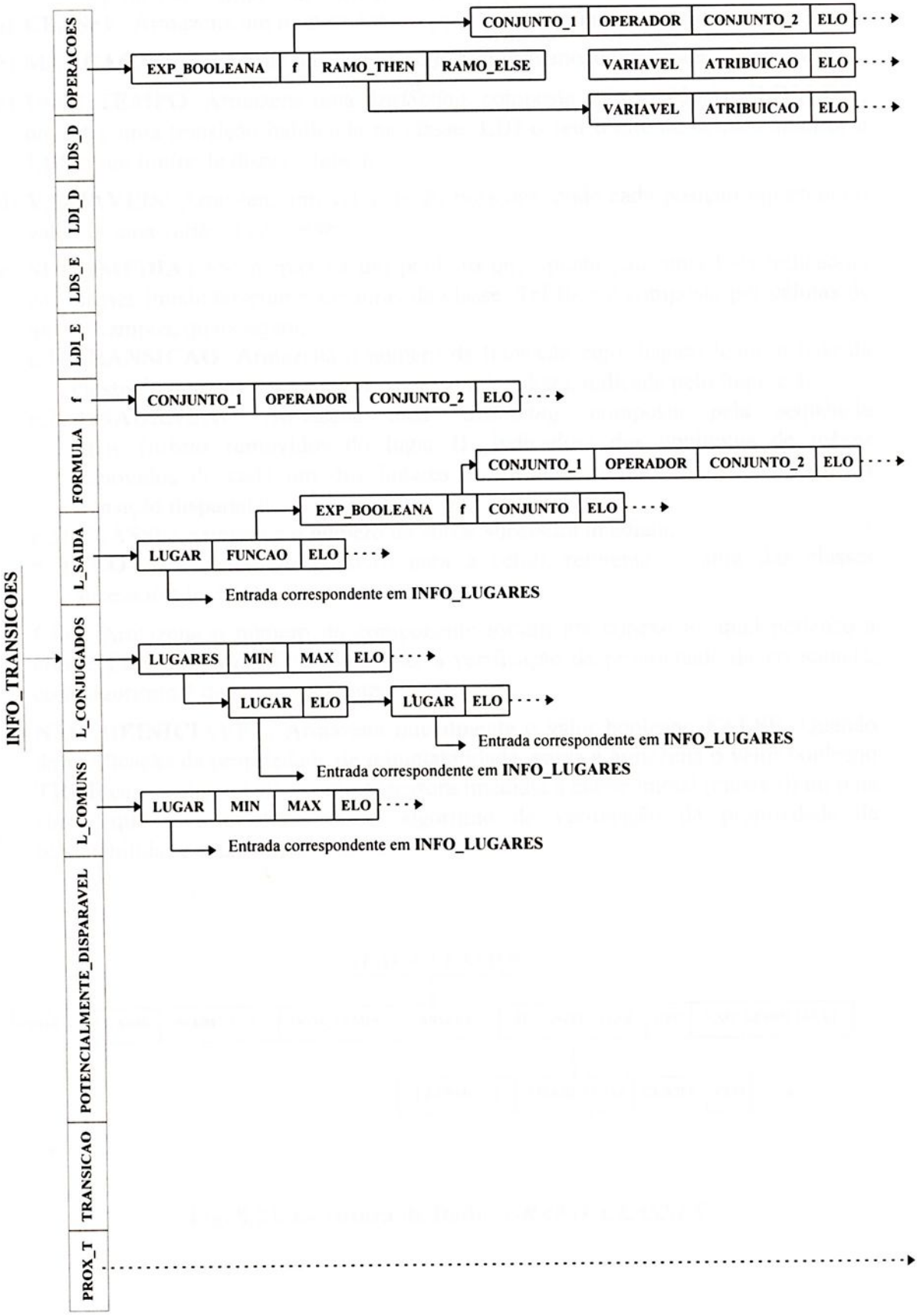


Fig. 8.22. Estrutura de Dados *INFO_TRANSICOES*.

Os principais campos da estrutura **GRAFO_CLASSES** (figura 8.23) são:

a) **CLASSE**: Armazena um número inteiro positivo, representando o número da classe;

b) **MARCACAO**: Armazena uma *AnsiString*, representante da marcação da classe;

c) **DOM_TEMPO**: Armazena uma *AnsiString*, composta pela seqüência **t[LDI,LDS]**, onde **t** é uma transição habilitada na classe, **LDI** o seu limite de disparo inferior e **LDS** o seu limite de disparo superior;

d) **VARIAVEIS**: Armazena um vetor de 26 posições, onde cada posição representa o valor de uma variável na classe;

e) **SUC_IMEDIATAS**: Armazena um ponteiro que aponta para uma lista indicadora das classes imediatamente sucessoras da classe. Tal lista é composta por células de quatro campos, quais sejam:
   e.1) **TRANSICAO**: Armazena o número da transição cujo disparo levou a rede da classe do item a) para a classe sucessora imediata, indicada pelo item e.3;
   e.2) **AMARRACAO**: Armazena uma *AnsiString*, composta pela seqüência L(**l**)←{*tokens* removidos do lugar **l**}, indicadora dos conjuntos de *tokens* removidos de cada um dos lugares de entrada, comuns ou conjugados, da transição disparada;
   e.3) **CLASSE**: Armazena o número da classe sucessora imediata;
   e.4) **ELO**: Armazena um ponteiro para a célula referente a outra das classes sucessoras imediatas da classe do item a);

f) **CFC**: Armazena o número do componente fortemente conexo ao qual pertence a classe. Este campo é utilizado durante a verificação da propriedade de vivacidade, cujo algoritmo é dado mais adiante;

g) **NÃO_REINICIAVEL**: Armazena inicialmente o valor booleano **FALSE**. Quando da verificação da propriedade de reiniciabilidade, passa a armazena o valor booleano **TRUE** caso a classe tenha como sucessora imediata a classe inicial (classe 0) ou uma classe que leve à classe 0. O algoritmo de verificação da propriedade de reiniciabilidade é dado mais adiante.

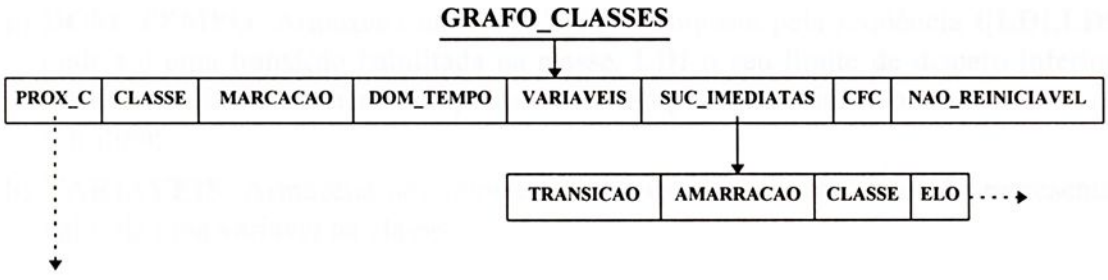


Fig. 8.23. Estrutura de Dados *GRAFO_CLASSES*.

A estrutura **SUCESSORAS_IMEDIATAS** (figura 8.24) é utilizada para guardar provisoriamente as informações referentes às classes sucessoras imediatas da classe que está sendo simulada. Pode-se observar na figura 8.24 a existência de três níveis do mesmo tipo de subestrutura. O primeiro nível refere-se a uma fila de classes cujas sucessoras imediatas devem ser calculadas. Tais sucessoras são armazenadas no segundo nível da estrutura. O terceiro nível representa a segunda geração de sucessoras do primeiro nível. Os campos encontram-se repetidos nestes três níveis com o intuito de uma maior rapidez no processo de simulação. Após a simulação da primeira classe da fila de classes do primeiro nível, ocorre:

1) Sendo a primeira célula do primeiro nível uma classe não pertencente à estrutura **GRAFO_CLASSES**, o conteúdo do seu campo **SUCESSORAS** é transferido para o campo **ELO** da última célula do primeiro nível e as informações da nova classe são inseridas em **GRAFO_CLASSES**;
2) A primeira célula do primeiro nível é removida da fila; e
3) Passa-se à simulação da nova primeira célula do primeiro nível.

O algoritmo de simulação completo será dado logo adiante.

Os principais campos da estrutura **SUCESSORAS_IMEDIATAS** são:

a) **SUCESSORAS**: Armazena um ponteiro que aponta para a fila de classes sucessoras imediatas da classe do nível a que pertence o campo. No último nível, assume o valor NIL, uma vez que as classes sucessoras do último nível só poderão ser calculadas quando as mesmas encontrarem-se no segundo nível;

b) **CLASSE**: Armazena um número inteiro positivo, representando o número da classe;

c) **TRANSICAO**: Armazena o número da transição cujo disparo levou a rede da classe antecessora para a classe sucessora. Útil somente nos níveis 2 e 3 das estruturas;

d) **AMARRACAO**: Armazena uma *AnsiString*, composta pela seqüência L(**l**)←{*tokens* removidos do lugar **l**}, indicadora dos conjuntos de *tokens* removidos de cada um dos lugares de entrada ou entrada conjugados da transição disparada. Útil somente nos níveis 2 e 3 da estrutura;

e) **TRANS_HAB**: Armazena um ponteiro que aponta para uma lista das transições habilitadas na classe;

f) **MARCACAO**: Armazena uma *AnsiString*, representante da marcação da classe;

g) **DOM_TEMPO**: Armazena uma *AnsiString*, composta pela seqüência **t[LDI,LDS]**, onde **t** é uma transição habilitada na classe, **LDI** o seu limite de disparo inferior e **LDS** o seu limite de disparo superior. Este campo só é definido nos níveis 1 e 2 da estrutura;

h) **VARIAVEIS**: Armazena um vetor de 26 posições, onde cada posição representa o valor de uma variável na classe;

Com base na descrição das principais estruturas de dados do Simulador RPTAN, serão dados, a seguir, o algoritmo de simulação de RPTAN e os algoritmos de verificação das propriedades de reiniciabilidade e vivacidade.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Fig. 8.24. Estrutura de Dados *CLASSES_PROVISORIAS*.

8.3.2. ALGORITMO DE SIMULAÇÃO

Esta subseção dará uma explanação geral do algoritmo de simulação implementado no Simulador RPTAN. Para os que quiserem uma compreensão mais detalhada deste algoritmo, recomenda-se o acompanhamento do fluxograma da figura 8.25.

Levando-se em conta as regras de funcionamento dadas no capítulo 7, seção 2, o algoritmo de simulação apresentado na figura 8.25 pode ser simplificado pelos seguintes passos:

1) Fazer inicializações;
2) Calcular, usando-se a lista **TOKENS_ATUAIS** de **INFO_LUGARES**, a marcação $\mathbf{M_0}$ da classe $\mathbf{C_0}$ e adicioná-la à definição da classe $\mathbf{C_0}$;
3) Inicializar **CLASSES_PROVISORIAS**;
4) Para cada transição de **INFO_TRANSICOES**, verificar as condições de habilitação em $\mathbf{C_0}$. Caso a transição for habilitada para alguma amarração, armazená-la na lista de transições habilitadas (**TRANS_HAB**) da primeira entrada do nível 1 da estrutura de **CLASSES_PROVISORIAS**;
5) Definir o domínio de tempo $\mathbf{D_0}$ da classe inicial $\mathbf{C_0}$ e inseri-lo na definição da classe. Inserir $\mathbf{C_0}$ na estrutura **GRAFO_CLASSES**. Caso $\mathbf{D_0}$ seja vazio, encerrar o algoritmo. Caso contrário, prosseguir;
6) Fazer $\mathbf{C_{corrente}}$ igual a $\mathbf{C_0}$;
7) Tomando a lista de transições habilitadas (**TRANS_HAB**) em $\mathbf{C_{corrente}}$, verificar as que estão de acordo com a regra de disparo;
8) Para cada transição **t** que estiver de acordo com a regra de disparo, calcular todas as amarrações possíveis e, para cada amarração α que torne habilitada a transição, efetuar as operações da transição e inserir a marcação $\mathbf{M_{t\alpha}}$ da classe decorrente de $\mathbf{C_{corrente}}$ pelo disparo de **t**, mediante a amarração α, na lista **SUCESSORAS** da primeira entrada do nível 1 de **CLASSES_PROVISORIAS**. Neste instante, as sucessoras imediatas $\mathbf{C_k}$ da classe $\mathbf{C_{corrente}}$ ainda não estão completamente definidas, pois falta a definição dos domínios de tempo das mesmas. Para definir esses domínios de tempo, deve-se continuar o algoritmo;
9) Para cada classe $\mathbf{C_k}$ pertencente à lista **SUCESSORAS** da primeira entrada do nível 1 da estrutura **CLASSES_PROVISORIAS**, fazer:
   9.1) Para cada transição de **INFO_TRANSICOES**, verificar as condições de habilitação na classe $\mathbf{C_k}$. Caso a transição seja habilitada para alguma amarração, armazená-la na lista de transições habilitadas da classe $\mathbf{C_k}$;
   9.2) Tomando a lista de transições habilitadas em $\mathbf{C_k}$, verificar as que estão de acordo com a regra de disparo;
   9.3) Para cada transição **s** que estiver de acordo com a regra de disparo, calcular todas as amarrações possíveis e, para cada amarração α que torne habilitada a transição, efetuar as operações da transição e inserir a marcação $\mathbf{M_{s\alpha}}$ da classe decorrente de $\mathbf{C_k}$ pelo disparo de **s** mediante a amarração α, na lista **SUCESSORAS** da entrada do nível 2 de **CLASSES_PROVISORIAS**, correspondente a $\mathbf{C_k}$. Neste instante, os domínios de tempo das sucessoras

imediatas C_k da classe $C_{corrente}$ estão definidos e, conseqüentemente, as classes C_k estão completamente definidas;

10) Para cada classe C_k (agora totalmente definida) pertencente à lista **SUCESSORAS** da primeira entrada do nível 1 da estrutura **CLASSES_PROVISORIAS**, verificar se a mesma é uma classe de **GRAFO_CLASSES**. São duas as possibilidades:
    a) Classe C_k pertence a **GRAFO_CLASSES**: Neste caso, faz-se a atualização do campo **SUC_IMEDIATAS** da classe correntemente simulada ($C_{corrente}$) e desaloca-se a classe C_k da lista **SUCESSORAS** da primeira entrada do nível 1 da estrutura **CLASSES_PROVISORIAS**;
    b) Classe C_k não pertence a **GRAFO_CLASSES**: Neste caso, insere-se as informações de C_k ao final de **GRAFO_CLASSES**, faz-se a atualização do campo **SUC_IMEDIATAS** da classe correntemente simulada ($C_{corrente}$) e desloca-se o ponteiro que aponta para a classe C_k da lista **SUCESSORAS** da primeira entrada do nível 1 da estrutura **CLASSES_PROVISORIAS** para o último campo **ELO** do nível 1 de **CLASSES_PROVISORIAS**;
11) Desalocar a primeira entrada do nível 1 da estrutura **CLASSES_PROVISORIAS**. Caso **CLASSES_PROVISORIAS** fique vazia, encerrar a simulação. Caso contrário, fazer $C_{corrente}$ igual à classe correspondente à primeira entrada do nível 1 da estrutura **CLASSES_PROVISORIAS** e repetir os passos do algoritmo a partir do passo 7.

Ao final da simulação, a estrutura **GRAFO_CLASSES** conterá as informações necessárias à construção do grafo de classes da rede.

Nas duas subseções seguintes, serão vistos os algoritmos de verificação das propriedades de reiniciabilidade e de vivacidade, os quais utilizam-se da estrutura **GRAFO_CLASSES** e do campo **POTENCIALMENTE_DISPARAVEL** de cada transição da estrutura **INFO_TRANSICOES**.

Inst. de Biociências, Letras e Ciências Exatas - UNESP
BIBLIOTECA
S. J. Rio Preto

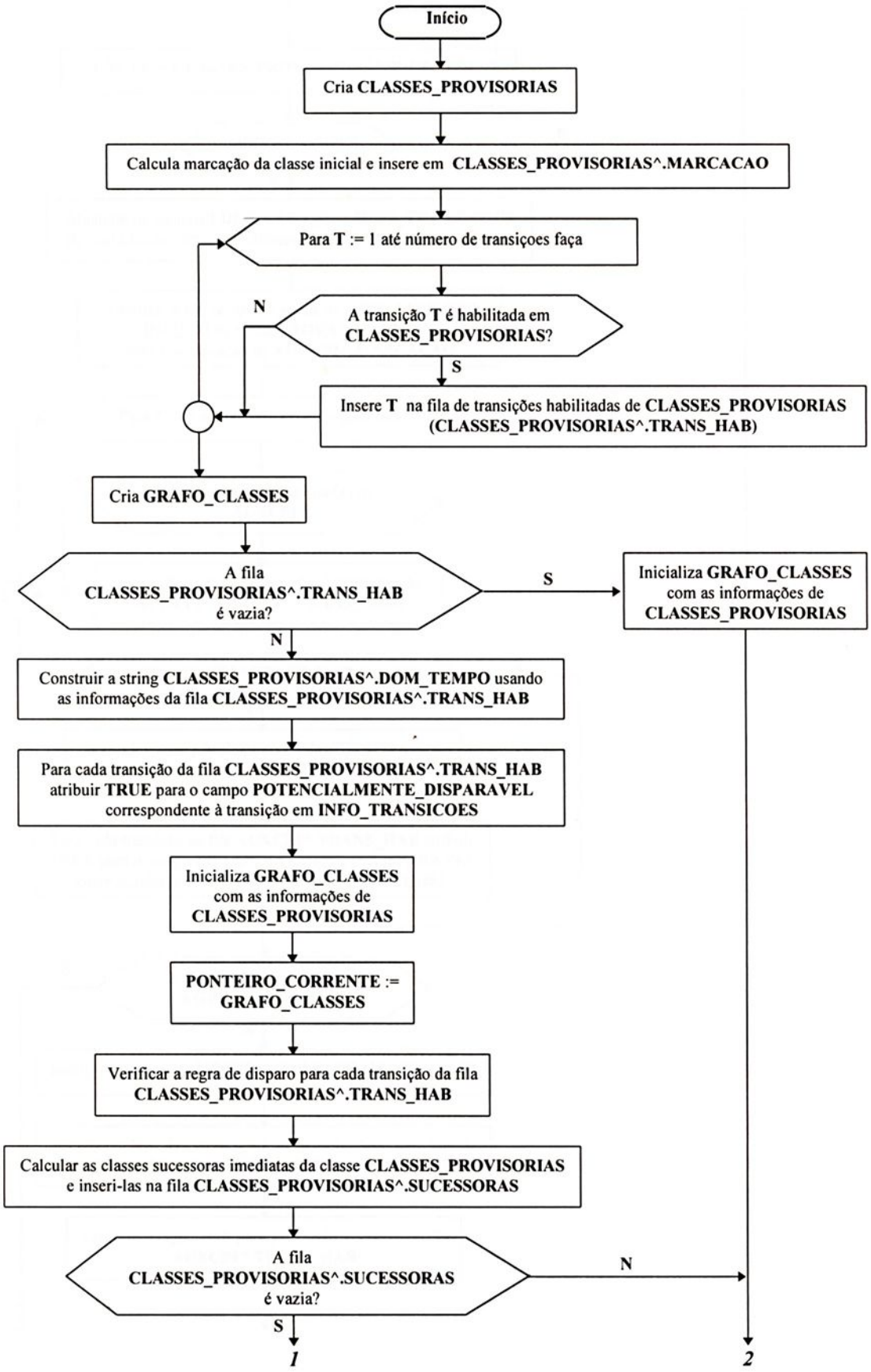
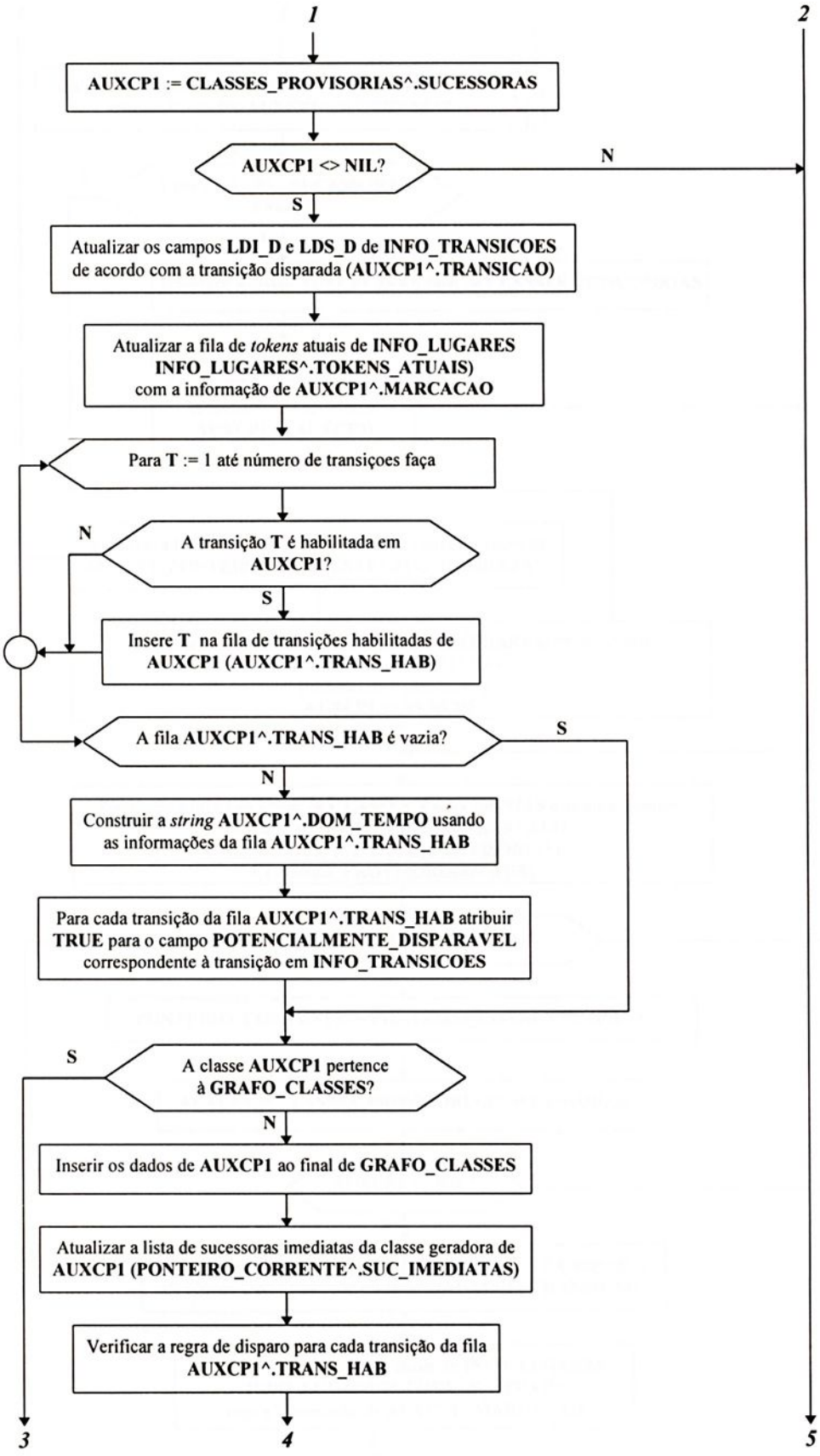
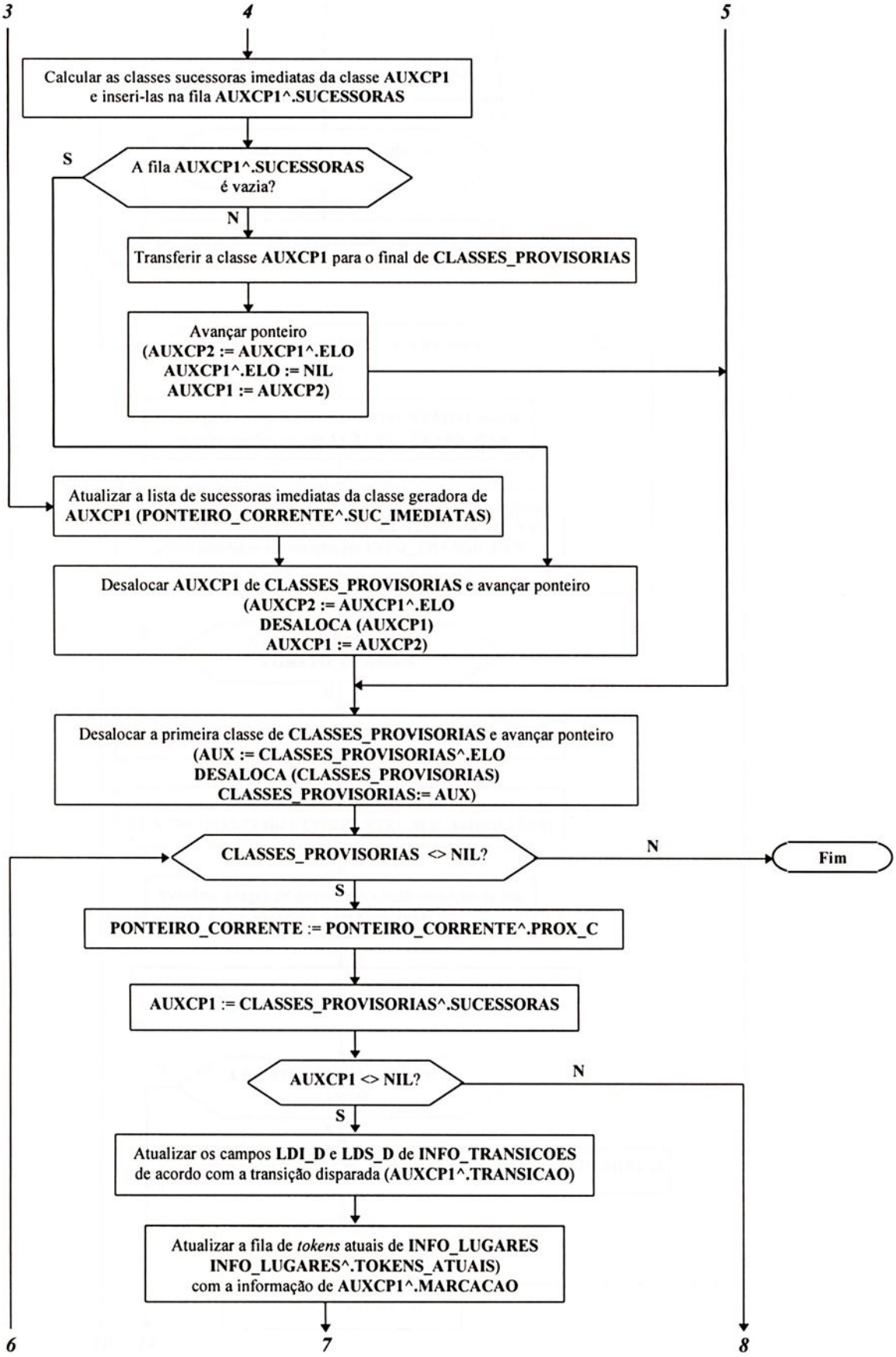


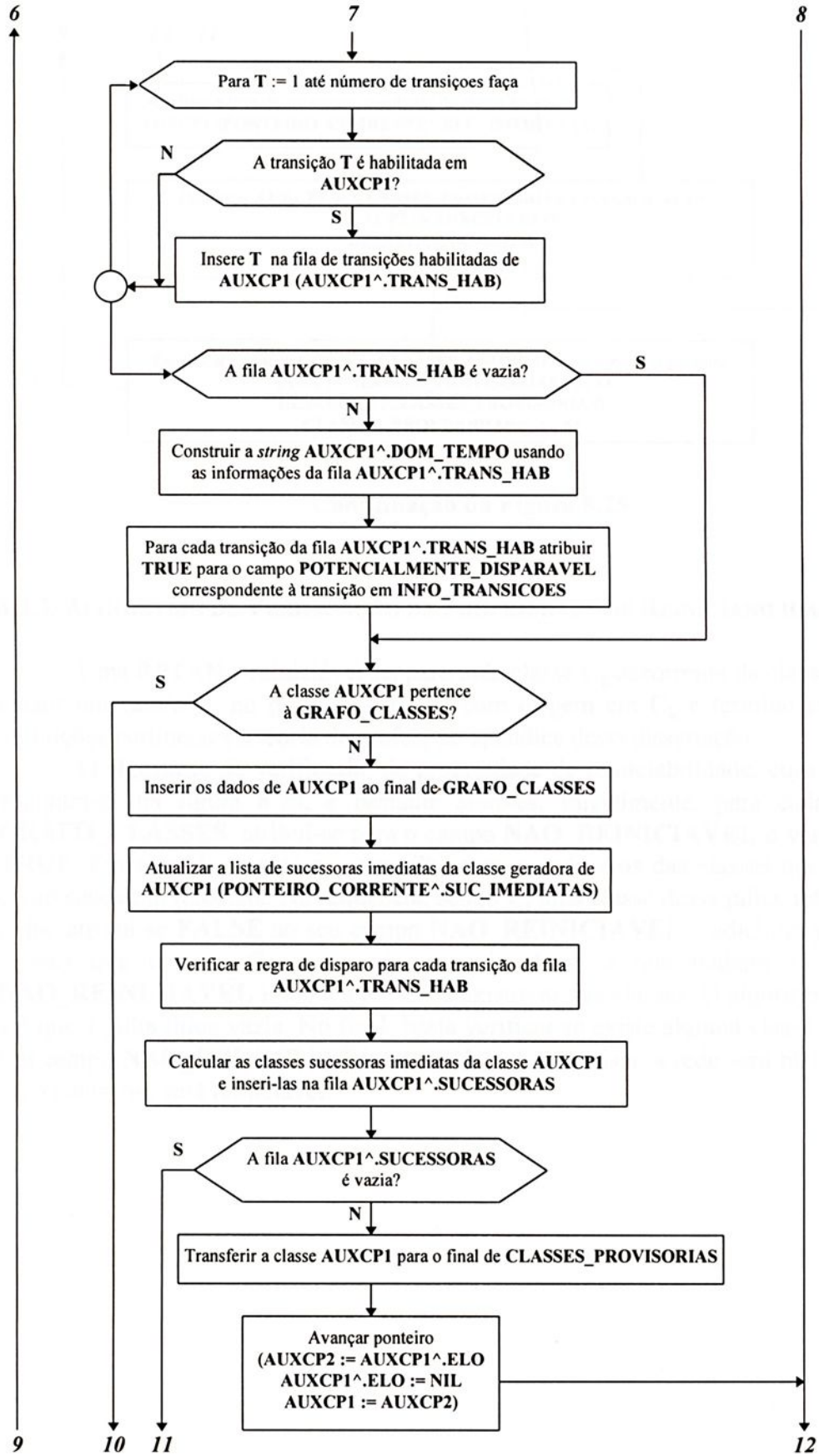
Fig. 8.25. Fluxograma do Algoritmo de Simulação de RPTAN.



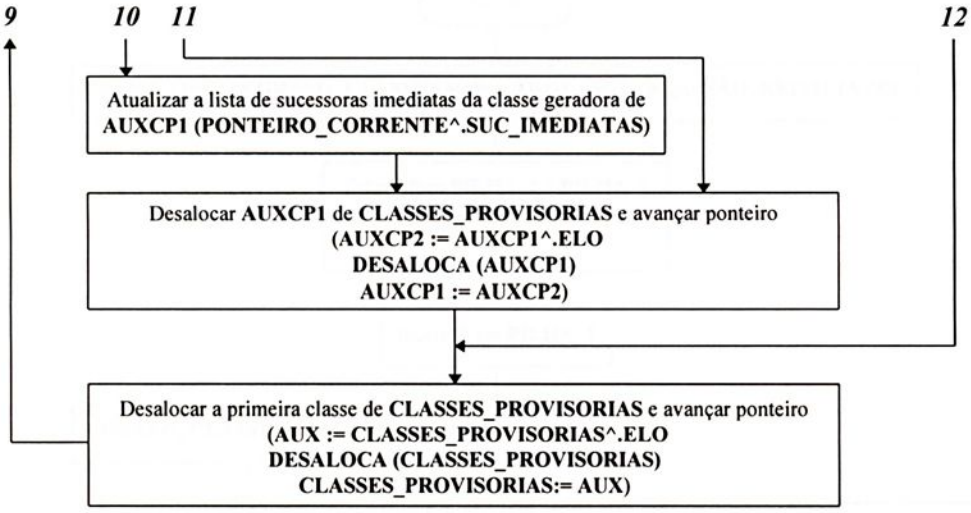
Continuação da Figura 8.25.



Continuação da Figura 8.25.



Continuação da Figura 8.25.



Continuação da Figura 8.25.

8.3.3. Algoritmo de Verificação da Propriedade de Reiniciabilidade

Uma RPTAN é reiniciável se, para toda classe $\mathbf{C_k}$ decorrente da classe inicial $\mathbf{C_0}$, existir um caminho, no grafo de classes, com origem em $\mathbf{C_k}$ e término em $\mathbf{C_0}$. Veja definições pertinentes à teoria de grafos, no apêndice desta dissertação.

O algoritmo de verificação da propriedade de reiniciabilidade, cujo fluxograma encontra-se na figura 8.26, é bastante simples. Inicialmente, para cada classe de **GRAFO_CLASSES**, atribui-se para o campo **NAO_REINICIAVEL** o valor booleano **TRUE**. Em seguida, monta-se uma pilha com os números das classes que tenham $\mathbf{C_0}$ como sucessora imediata. Na seqüência, sendo $\mathbf{C_r}$ uma classe dessa pilha, retira-se $\mathbf{C_r}$ da pilha, atribui-se **FALSE** ao seu campo **NAO_REINICIAVEL** e adiciona-se à pilha as classes que tenham $\mathbf{C_r}$ como sucessora imediata, e que tenham o seu campo **NAO_REINICIAVEL** igual a **TRUE**, se existirem tais classes. O algoritmo prossegue até que a pilha fique vazia. No final, basta verificar se existe alguma classe que tenha o seu campo **NAO_REINICIAVEL** em **TRUE**. Caso exista, a rede será não reiniciável; caso contrário, será reiniciável.

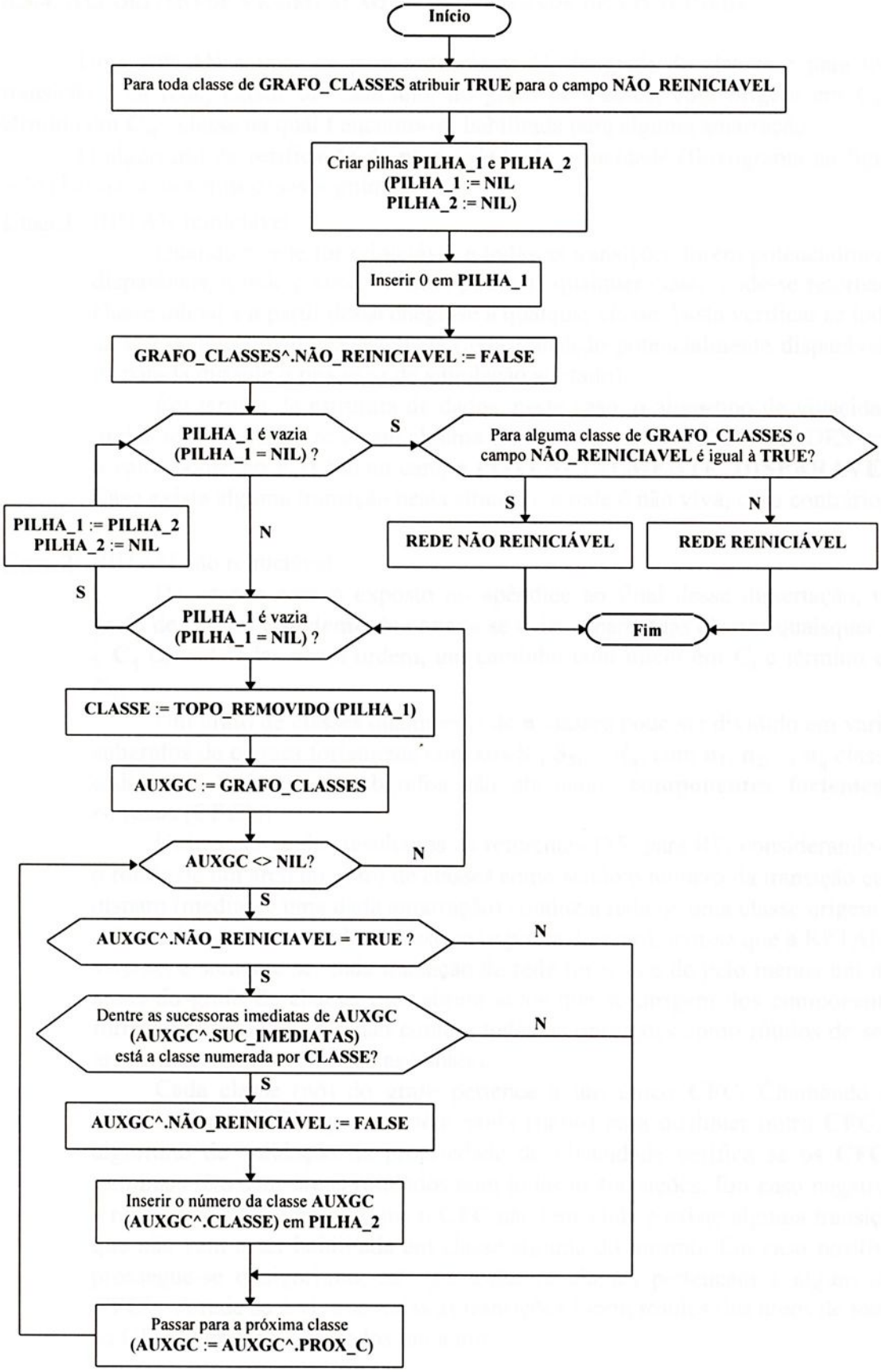


Fig. 8.26. Fluxograma do Algoritmo de Verificação de Reiniciabilidade.

8.3.4. ALGORITMO DE VERIFICAÇÃO DA PROPRIEDADE DE VIVACIDADE

Uma RPTAN é viva se, para toda classe $\mathbf{C_k}$ do grafo de classes e para toda transição **t** da rede, existir um caminho, no grafo de classes, com origem em $\mathbf{C_k}$ e término em $\mathbf{C_m}$, classe na qual **t** encontra-se habilitada para alguma amarração.

O algoritmo de verificação da propriedade de vivacidade (fluxograma na figura 8.26) baseia-se nos dois casos seguintes:

Caso 1: RPTAN reiniciável.

Quando a rede for reiniciável e todas as transições forem potencialmente disparáveis, a rede é viva. Como a partir de qualquer classe pode-se retornar à classe inicial e a partir dessa chega-se a qualquer classe, basta verificar se todas as são potencialmente disparáveis (toda transição potencialmente disparável é disparada durante o processo de simulação adotado).

Em termos da estrutura de dados, neste caso, o algoritmo de vivacidade limita-se a pesquisar se existe alguma transição de **INFO_TRANSICOES** com o valor booleano **FALSE** no campo **POTENCIALMENTE_DISPARAVEL**. Caso exista alguma transição nesta situação, a rede é não viva; caso contrário, é viva.

Caso 2: RPTAN não reiniciável.

De acordo com o exposto no apêndice ao final dessa dissertação, um grafo de classes é **fortemente conexo** se existir, para duas classes quaisquer $\mathbf{C_i}$ e $\mathbf{C_j}$ consideradas nessa ordem, um caminho com início em $\mathbf{C_i}$ e término em $\mathbf{C_j}$.

Um grafo de classes qualquer **G** de **n** classes pode ser dividido em vários subgrafos de classes fortemente conexos $\mathbf{S_1}$, $\mathbf{S_2}$,..., $\mathbf{S_q}$, com $\mathbf{n_1}$, $\mathbf{n_2}$,..., $\mathbf{n_q}$ classes cada um. Assim, tais subgrafos são chamados **componentes fortemente conexos** (**CFC**'s).

Utilizando-se dos resultados da referência [25] para RP, considerando-se o rótulo de um arco do grafo de classes como sendo o número da transição cujo disparo (mediante uma dada amarração) conduz a rede de uma classe origem (a origem do arco) a uma classe destino (a ponta do arco), tem-se que a RPTAN é viva se, e somente se, cada transição da rede for rótulo de pelo menos um dos arcos do grafo de classes e existirem arcos que se dirigem dos componentes fortemente conexos, que não contém todas as transições como rótulos de seus arcos, aos componentes que as contém.

Cada classe (nó) do grafo pertence a um único **CFC**. Chamando de **terminais** os **CFC**'s que não têm saída (ramo) para qualquer outro **CFC**, o algoritmo de validação da propriedade de vivacidade verifica se os **CFC**'s terminais têm seus arcos rotulados com todas as transições. Em caso negativo, a rede é não viva, uma vez que o **CFC** não tem saída e existe alguma transição que não vem a ser habilitada em classe alguma do mesmo. Em caso positivo, prossegue-se o algoritmo, até que todas as classes pertençam a algum dos **CFC**'s. A rede só é viva se todas as transições forem rótulos dos arcos de todos os **CFC**'s terminais, tomados um a um.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

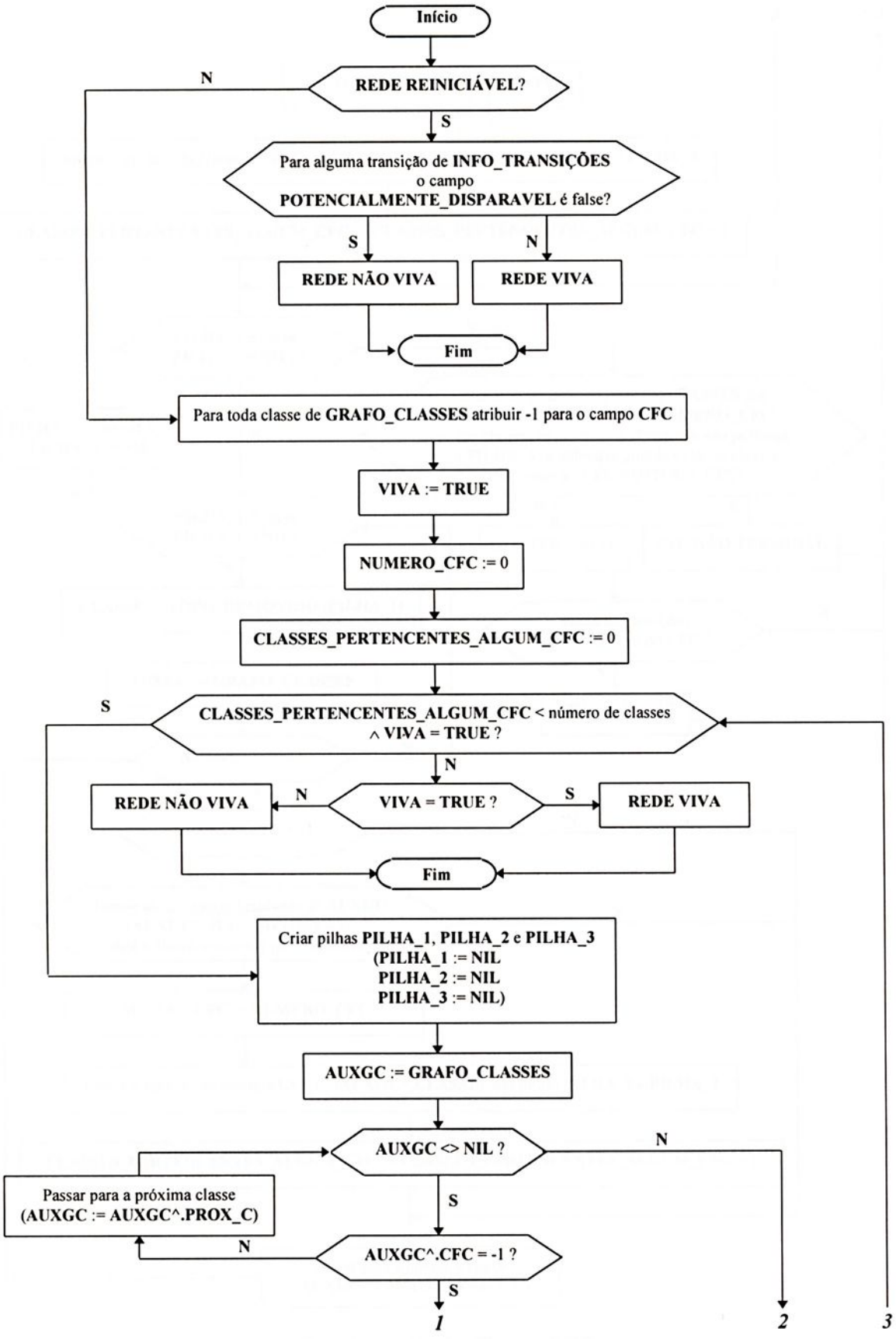
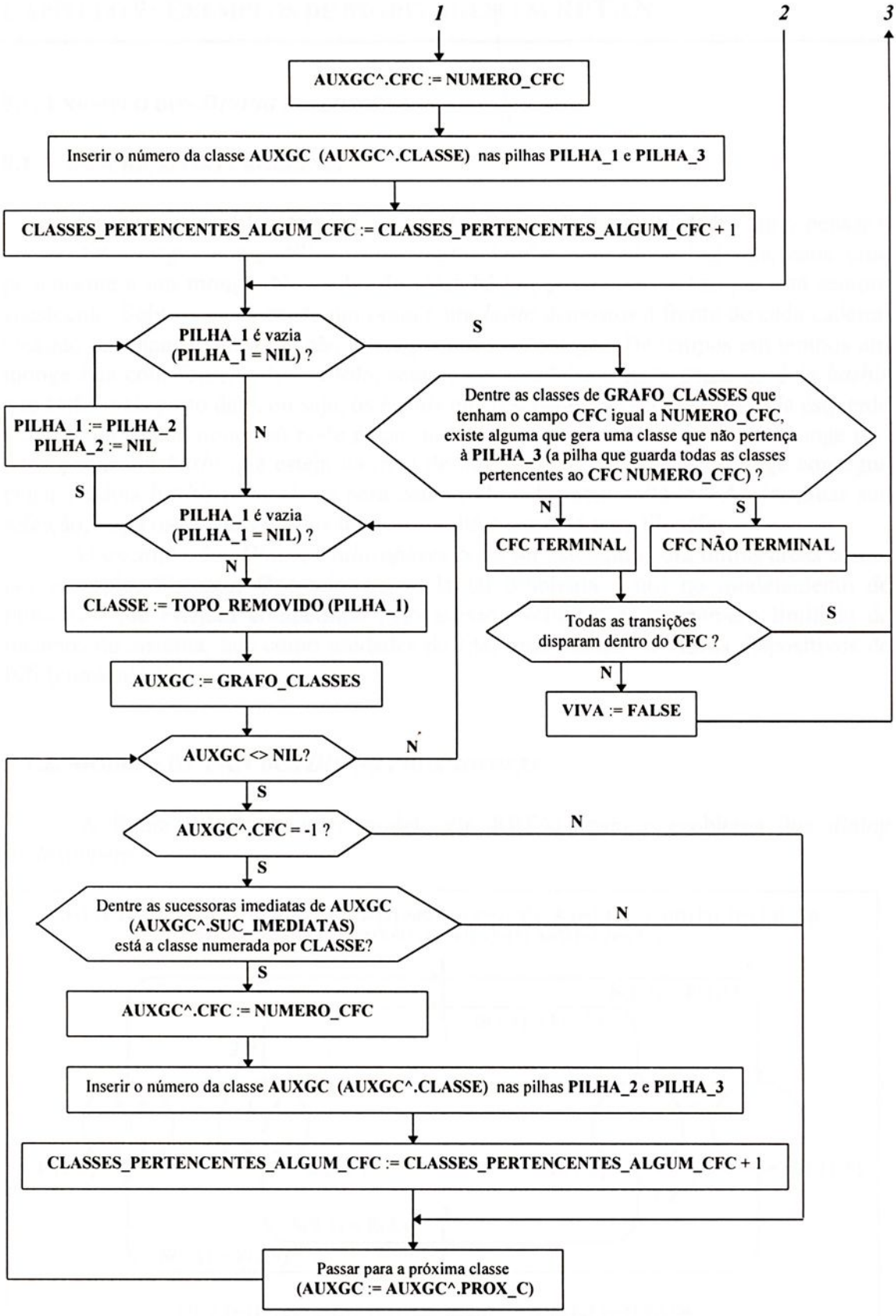


Fig. 8.27. Fluxograma do Algoritmo de Verificação de Vivacidade.



Continuação da Figura 8.27.

CAPÍTULO 9: EXEMPLOS DE MODELAGEM EM RPTAN

9.1. EXEMPLO DOS *DINING PHILOSOPHERS*

9.1.1. DESCRIÇÃO DO PROBLEMA

Cinco monges filósofos passam a vida alternando suas atividades entre pensar e comer. Os monges compartilham uma mesa circular com cinco cadeiras, cada uma pertencente a um monge. No centro da mesa há um prato com arroz, que está sempre abastecido. Sobre a mesa existe um prato e um *hashi* dispostos à frente de cada cadeira. Quando um monge pensa ele não interage com seus colegas. De tempos em tempos um monge fica com fome, entra na sala, senta na sua cadeira e tenta pegar os dois *hashis* que estão mais perto dele, ou seja, os *hashis* que o separam de seus vizinhos da esquerda e da direita. Cada monge só pode pegar, por vez, um *hashi*. Obviamente, o monge não pode pegar um *hashi* que esteja na mão de um vizinho. Quando um monge consegue pegar os dois *hashis* necessários para comer, ele o faz sem soltá-los. Ao terminar sua refeição, larga os *hashis* em seus lugares e volta à sua cela para filosofar.

O exemplo dos *Dining Philosophers* pode ser estendido para outras áreas como, por exemplo, Sistemas Operacionais, onde tal problema é útil no modelamento de processos que estejam competindo pelo acesso exclusivo a um número limitado de recursos do sistema, tais como unidades de fita, impressoras ou outros dispositivos de E/S (entrada/saída).

9.1.2. MODELO RPTAN DOS *DINING PHILOSOPHERS*

A figura 9.1 descreve o modelo em RPTAN para o problema dos *dining philosophers*.

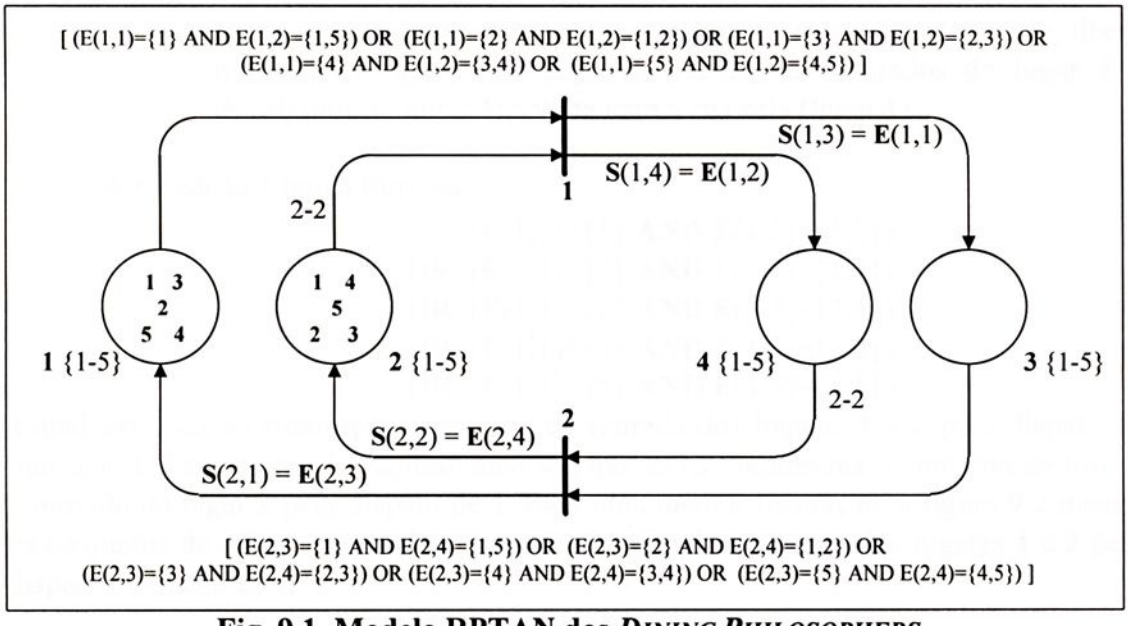


Fig. 9.1. Modelo RPTAN dos *DINING PHILOSOPHERS*.

Na rede da figura 9.1, os lugares têm o seguinte significado:

a) Lugar **1**: **Monges filosofando**.
Os *tokens* presentes neste lugar indicam os monges que estão filosofando. Cada monge é associado à um único tipo de *token*;

b) Lugar **2**: ***Hashis* disponíveis**.
Os *tokens* presentes neste lugar indicam os *hashis* que estão disponíveis sobre a mesa. Cada *hashi* é associado à um único tipo de *token*;

c) Lugar **3**: **Monges jantando**.
Os *tokens* presentes neste lugar indicam os monges que estão jantando. Usa-se a associação monge-*token* dada no lugar **1**;

d) Lugar **4**: ***Hashis* ocupados**.
Os *tokens* presentes neste lugar indicam os *hashis* que estão na mão de algum monge. Usa-se a associação *hashi-token* dada no lugar **2**.

Para não haver situações de *deadlock* (sistema travado pela falta de ações possíveis), a RPTAN da figura 9.1 adotou o critério de que o monge deve pegar os dois *hashis* de que necessita de uma só vez. Caso um dos *hashis* não esteja disponível, o monge não poderá pegar o outro, mesmo que esteja disponível. Esta abordagem é adotada em Sistemas Operacionais onde, primeiro, verifica-se a disponibilidade dos recursos necessários a um processo, antes que o mesmo apreenda para si um recurso que possa ser necessário a outro processo. Assim, as transições da rede da figura 9.1 têm o seguinte significado:

a) Transição **1**: **Monge pega 2 *hashis* e começa a comer**.
Um dos monges que estavam filosofando (lugar **1**), pega os 2 *hashis* a que tem direito (observe o rótulo 2-2 do arco e a fórmula da transição) e começa a comer (lugar **3**). Os *hashis* pegos são retirados do lugar **2** e repassados para o lugar **4**;

b) Transição **2**: **Monge deixa de comer e disponibiliza os 2 *hashis* que havia pego**.
Um dos monges que estavam jantando (lugar **3**) deixa de comer, libera os 2 *hashis* que havia pego (os *hashis* são retirados do lugar **4** e devolvidos ao lugar **2**) e volta para a sua cela (lugar **1**).

A transição **1** tem a fórmula

$$\begin{aligned}&(\mathbf{E}(1,1)=\{1\}\ \mathbf{AND}\ \mathbf{E}(1,2)=\{1,5\})\\ \mathbf{OR}\ &(\mathbf{E}(1,1)=\{2\}\ \mathbf{AND}\ \mathbf{E}(1,2)=\{1,2\})\\ \mathbf{OR}\ &(\mathbf{E}(1,1)=\{3\}\ \mathbf{AND}\ \mathbf{E}(1,2)=\{2,3\})\\ \mathbf{OR}\ &(\mathbf{E}(1,1)=\{4\}\ \mathbf{AND}\ \mathbf{E}(1,2)=\{3,4\})\\ \mathbf{OR}\ &(\mathbf{E}(1,1)=\{5\}\ \mathbf{AND}\ \mathbf{E}(1,2)=\{4,5\}),\end{aligned}$$

a qual expressa as amarrações passíveis de retirada dos lugares **1** e **2** pelo disparo da transição **1**. Lembre-se, do capítulo anterior, que **E**(1,2) representa o conjunto de *tokens* removido do lugar **2** pelo disparo de **1**. Para uma melhor ilustração, a figura 9.2 mostra os conjuntos de *tokens* que podem ser retirados simultaneamente dos lugares **1** e **2** pelo disparo da transição **1**.

unesp

A fórmula da transição **2** tem é análoga à fórmula da transição **1**, bastando substituir-se o lugar **1** por **3** e o lugar **2** por **4**.

	Lugar 1	Lugar 2	Significado no Sistema Modelado
Conjuntos Retirados	{1}	{1,5}	Monge 1 só pode pegar hashis 1 e 5
	{2}	{1,2}	Monge 2 só pode pegar hashis 1 e 2
	{3}	{2,3}	Monge 3 só pode pegar hashis 2 e 3
	{4}	{3,4}	Monge 4 só pode pegar hashis 3 e 4
	{5}	{4,5}	Monge 5 só pode pegar hashis 4 e 5

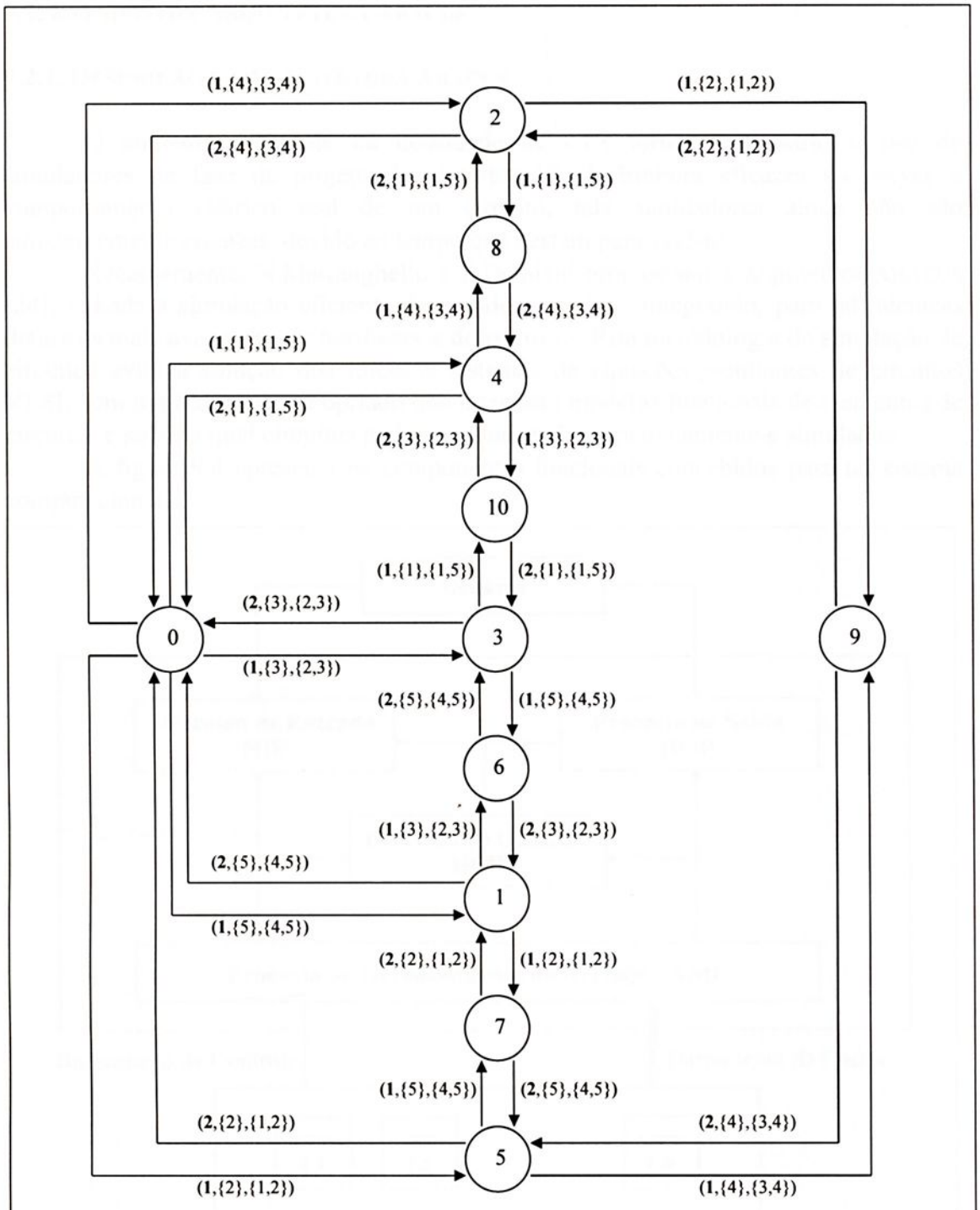
Fig. 9.2. Conjuntos de Tokens que podem ser Retirados Simultaneamente dos Lugares 1 e 2 pelo Disparo da Transição 1 da Figura 9.1.

Terminando as considerações referentes à RPTAN da figura 9.1, observa-se, por exemplo, que o ramo de saída ligando a transição **1** ao lugar **4** tem a função de saída de *tokens*:

$$\mathbf{S}(1,4) = \mathbf{E}(1,2),$$

significando que, os *hashis* retirados do lugar **2** (ou seja, **E**(1,2)) serão acrescidos ao lugar **4**, quando do disparo da transição **1**. As demais funções de saída de *tokens* têm comportamento análogo à da função **S**(1,4).

O grafo de classes fornecido pelo Simulador RPTAN para a rede da figura 9.1 encontra-se representado na figura 9.3. O simulador gerou 11 classes, além de verificar que a rede é viva e reiniciável. Observe que, na classe **0**, todos os monges estão filosofando (lugar **1** com *tokens* de **1** a **5**). Nas classes de **1** a **5**, há apenas um monge jantando (lugar **3** possui apenas um *token*). Já nas classes de **6** a **10**, há dois monges jantando simultaneamente (lugar **3** possui dois *tokens*). Verifica-se, assim, que o número máximo de monges jantando ao mesmo tempo é dois, devido à não disponibilidade dos *hashis* necessários aos demais monges. Desta forma, a RPTAN da figura 9.1 modela adequadamente o exemplo dos *dining philosophers*.



lasse	Marcação {(Lugar;Tokens ...), ...}	Dom. Tempo {(Trans; LDI, LDS), ...}	lass	Marcação {(Lugar;Tokens...), ...}	Dom. Tempo {(Trans; LDI, LDS), ...}
0	{(1;1,2,3,4,5), (2;1,2,3,4,5)}	{(1;0,0)}	6	{(1;1,2,4), (2;1),(3;3,5),(4;2,3,4,5)}	{(2;0,0)}
1	{(1;1,2,3,4), (2;1,2,3),(3;5),(4;4,5)}	{(1;0,0), (2;0,0)}	7	{(1;1,3,4), (2;3),(3;2,5),(4;1,2,4,5)}	{(2;0,0)}
2	{(1;1,2,3,5), (2;1,2,5),(3;4),(4;3,4)}	{(1;0,0), (2;0,0)}	8	{(1;2,3,5), (2;2),(3;1,4),(4;1,3,4,5)}	{(2;0,0)}
3	{(1;1,2,4,5), (2;1,4,5),(3;3),(4;2,3)}	{(1;0,0), (2;0,0)}	9	{(1;1,3,5), (2;5),(3;2,4),(4;1,2,3,4)}	{(2;0,0)}
4	{(1;2,3,4,5), (2;2,3,4),(3;1),(4;1,5)}	{(1;0,0), (2;0,0)}	10	{(1;2,4,5), (2;4),(3;1,3),(4;1,2,3,5)}	{(2;0,0)}
5	{(1;1,3,4,5), (2;3,4,5),(3;2),(4;1,2)}	{(1;0,0), (2;0,0)}			

Fig. 9.3. Grafo de Classes da RPTAN da Figura 9.1.

9.2. EXEMPLO DA ARQUITETURA ABACUS

9.2.1. DESCRIÇÃO DA ARQUITETURA ABACUS

O aumento crescente na densidade de CI's tornou necessário o uso de simuladores na fase de projeto de circuitos VLSI. Embora eficazes ao prever o comportamento elétrico real de um circuito, tais simuladores ainda não são suficientemente eficazes, devido ao tempo que gastam para fazê-lo.

Recentemente, N.Marranghello e F.Damiani propuseram a arquitetura ABACUS [24], voltada à simulação eficiente de grandes circuitos, integrando, para tal, técnicas dentre as mais avançadas de *hardware* e de *software*. Esta metodologia de simulação de circuitos evita a solução dos imensos sistemas de equações resultantes de circuitos VLSI, com um *hardware* apropriado que incorpora modelos funcionais de elementos de circuitos e sobre o qual circuitos podem ser mapeados dinamicamente e simulados.

A figura 9.4 apresenta os componentes funcionais concebidos para tal sistema computacional.

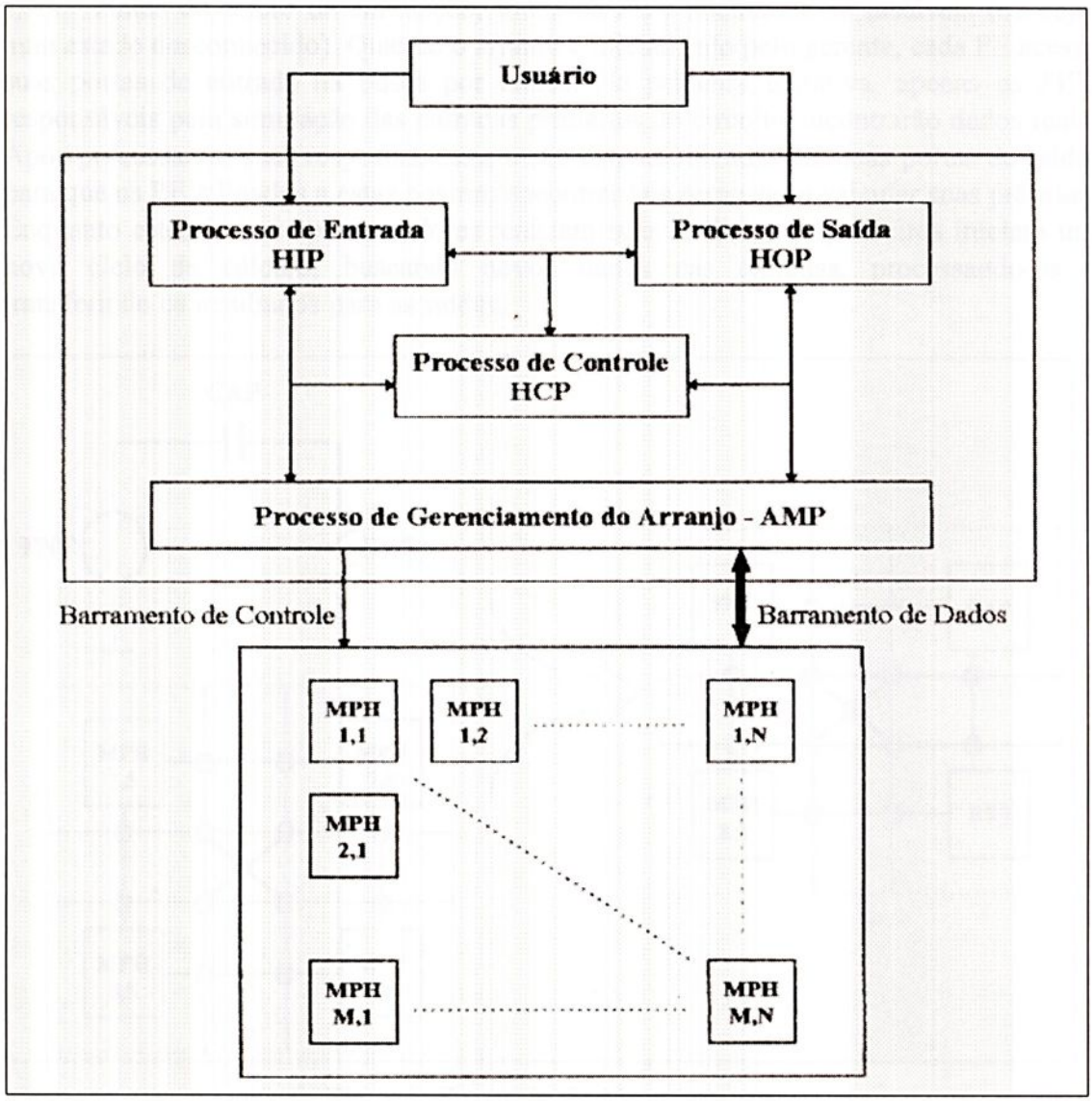


Fig. 9.4. Interação Entre os Componentes da Arquitetura ABACUS.

A arquitetura ABACUS possui um processador mais complexo, o qual atua como **gerente** (hospedeiro) de um arranjo, e um conjunto de processadores mais simples, cada um dos quais dedicados à tarefa de simular o modelo de um componente de um dado circuito.

Os processadores do arranjo, chamados **processadores especiais** (PE), trabalham de forma assíncrona entre si, interagindo, quando necessário, de acordo com um princípio de operação de fluxo de dados. A interconexão entre tais processadores é reconfigurável dinamicamente pelo hospedeiro, a fim de refletir a topologia do circuito que está sendo simulado.

O mapeamento de circuitos nos PE's é exemplificado na figura 9.5. Nela, foi mapeado um circuito RC passa-altas numa parte do arranjo com 4 processadores. Para tal, precisou-se encontrar 3 processadores desocupados que pudessem ser ligados entre si. Assim, os processadores <7,5>, <7,6> e <8,6> do arranjo foram alocados com os elementos **FNT**, **CAP** e **RES**, respectivamente a fonte, o capacitor e o resistor. O mapeamento foi concretizado através do fechamento das chaves que ligam os processadores correspondentes, conforme destacado na figura 9.5.

Antes do início da simulação, todos os PE's encontram-se instáveis (ou seja, num estado desconhecido). Quando o arranjo é inicializado pelo gerente, cada PE acessa suas portas de entrada na busca por dados. Na primeira tentativa, apenas os PE's responsáveis pela simulação das entradas primárias do circuito encontrarão dados reais. Após processarem suas respostas, estas serão disponibilizadas em suas portas de saída, para que os PE's ligados a estas possam encontrar tais respostas e calcular suas próprias. Enquanto estes últimos processadores realizam suas análises, os primeiros iniciam um novo ciclo de cálculo, buscando novos dados nas entradas, processando-os e transferindo os resultados para as saídas.

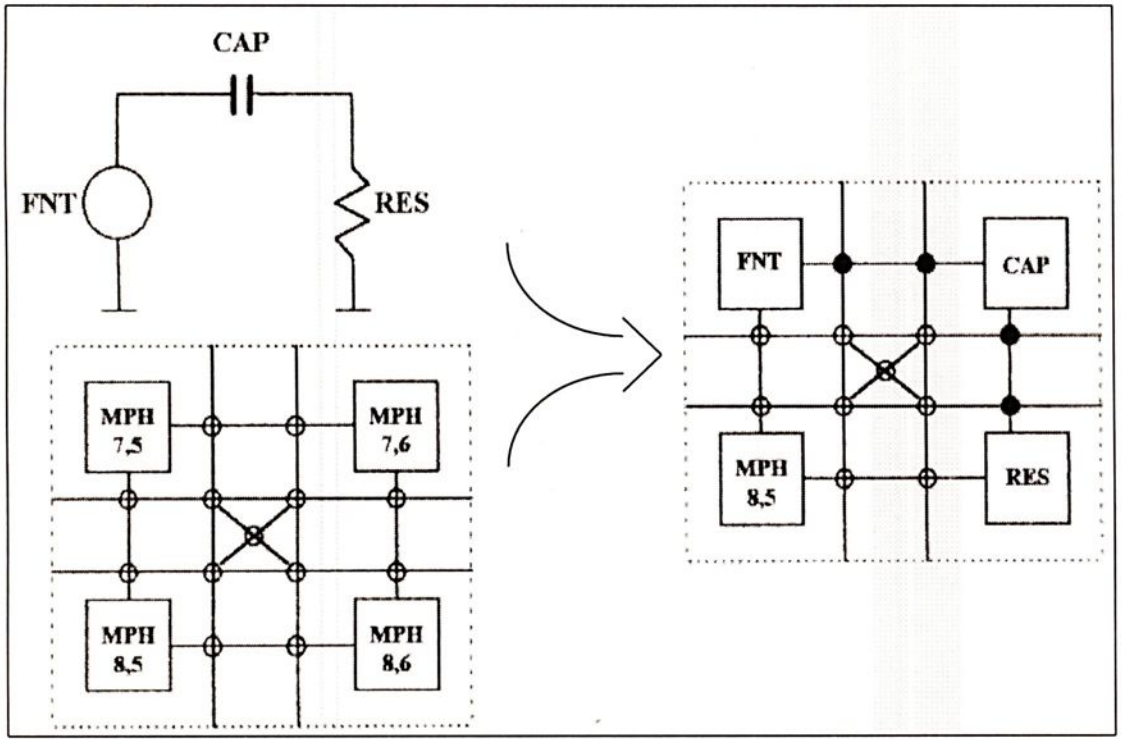


Fig. 9.5. Exemplo de Mapeamento.

Um instante de simulação é concluído quando todos os PE's convergirem para uma solução. Neste ponto, os dados são transferidos para o gerente, o qual irá liberar oportunamente o início da simulação do instante seguinte, se for o caso de fazê-lo.

Destarte, o circuito é analisado como se uma fotografia de seu funcionamento fosse tirada a cada instante de simulação.

Uma referência completa sobre o funcionamento de um PE encontra-se na referência [24].

O algoritmo de simulação usado na arquitetura ABACUS pode ser representado através de cinco processos principais, quatro dos quais seriam executados no processador hospedeiro e o outro executado em cada um dos processadores do arranjo (vide figura 9.4). Estes processos são os seguintes:

a) **HCP**: Processo que serve para controlar a operação do sistema como um todo, iniciando e encerrando a operação do ambiente de simulação e ativando e desativando cada um dos demais processos do hospedeiro, sempre que se fizer necessário. Assim sendo, tal controle deve ser entendido como um gerenciamento assíncrono dos demais processos do hospedeiro;
b) **HIP**: Processo que tem por finalidade processar os dados de entrada fornecidos pelo usuário, formatando-os para o processamento pelos demais processos do simulador;
c) **HOP**: Processo destinado ao fornecimento das respostas requisitadas pelo usuário, o que é feito a partir de resultados de simulação e da base de dados formatada pelo processo **HIP**;
d) **AMP**: Processo de controle do funcionamento do arranjo de processadores, constituindo-se num ponto onde, após o término de suas tarefas, os processadores do arranjo se encontram para receber outras tarefas;
e) **MPH**: Processo que, ao contrário dos anteriores, roda em cada um dos PE's. Trata-se do processo responsável pela execução efetiva da simulação dos componentes do circuito, com base nos dados fornecidos pelo processo **AMP**.

Resumindo, em ABACUS, existem vários processadores onde se encontram os modelos que representam o comportamento dos diversos elementos do circuito. Estes processadores (denominados por PE) são interconectados através de um conjunto de barramentos programáveis, como se o circuito a ser simulado fosse interconectado em uma *protoboard*. A seguir, os dados primários do circuito são fornecidos ao arranjo e a simulação é iniciada. A partir do estado inicial, todos os processadores do arranjo começam a trabalhar simultaneamente, buscando dados em suas filas de entrada, processando-os e transferindo os resultados às suas filas de saída. Inicialmente, como num circuito real, apenas os processadores ligados às entradas primárias do circuito gerarão sinais reais. Conforme novas iterações são realizadas, estes sinais reais propagam-se pelo circuito (arranjo), até alcançarem as saídas. Este processo continua até que todo o arranjo se estabilize para um dado conjunto de sinais de entrada. Neste momento, as soluções são armazenadas, o tempo é incrementado e um novo conjunto de entradas primárias é aplicado ao circuito. O processo descrito neste parágrafo é repetido até que todos os instantes desejados sejam analisados.

Na arquitetura ABACUS, o paralelismo é implementado na sua forma mais ampla. Apenas os instantes inicial e final do funcionamento dos processadores do arranjo são determinados pelo processador hospedeiro, mas durante a execução da simulação, os

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

PE's funcionam concorrentemente e de forma totalmente assíncrona, tanto em relação ao hospedeiro, quanto entre si.

9.2.2. Modelo RPTAN da Arquitetura Abacus

Conforme explicado na introdução desta dissertação, o trabalho de mestrado atual originou-se a partir da necessidade de se simular a arquitetura Abacus em um PC.

A figura 9.6 descreve o funcionamento de uma arquitetura Abacus composta de 2 PE's, em RPTAN. Este pequeno número de processadores foi adotado com a finalidade de facilitar a compreensão do modelo.

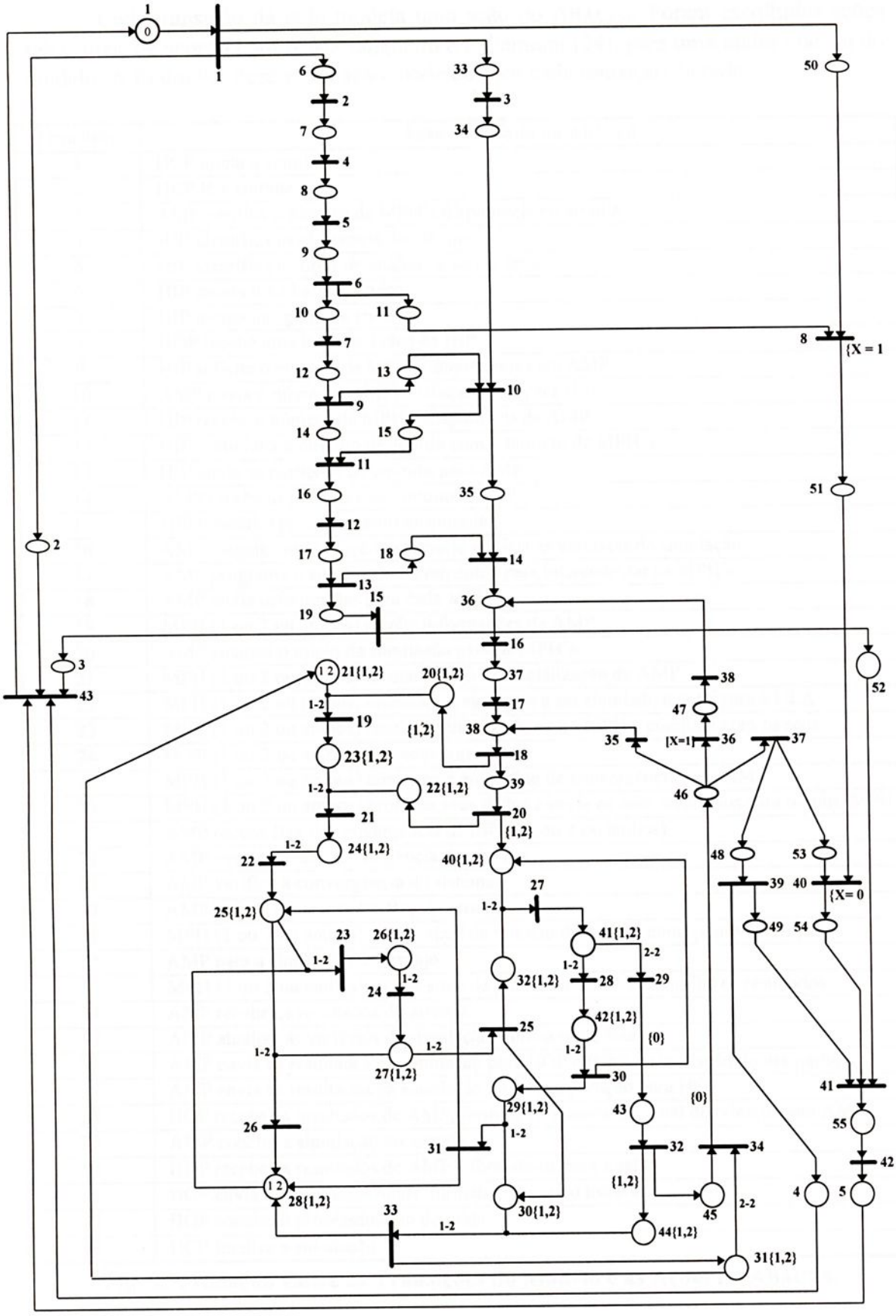


Fig. 9.6. Modelo RPTAN da Arquitetura ABACUS.

Cada transição da rede modela uma ação no ABACUS. Foram escolhidas ações relevantes, de acordo com N.Marranghello e F.Damiani [24], para uma maior clareza do modelo. A figura 9.7 descreve a ação modelada por cada transição da rede.

Transição	Ação Modelada no ABACUS
1	HCP inicia a simulação
2	HCP lê a entrada
3	AMP verifica o número de MPH's disponíveis no arranjo
4	HIP identifica os elementos do circuito
5	HIP identifica os tipos de análises a serem feitas
6	HIP monta uma base de dados
7	HIP monta um grafo de conexão
8	HOP recebe uma base de dados de HIP
9	HIP solicita o número de MPH's disponíveis para AMP
10	AMP envia o número de MPH's disponíveis para HIP
11	HIP recebe o número de MPH's disponíveis de AMP
12	HIP particiona o circuito de acordo com o número de MPH's
13	HIP envia as partições do circuito para AMP
14	AMP recebe as partições do circuito de HIP
15	HIP conclui o processamento de entrada
16	AMP mapeia uma partição no arranjo e inicia as variáveis de simulação
17	AMP programa o sistema de barramentos para interconectar os MPH's
18	AMP envia informações para cada MPH
19	MPH (1 ou 2 ou ambos) recebe informações de AMP
20	AMP sinaliza o início da simulação para os MPH's
21	MPH (1 ou 2 ou ambos) recebe o sinal de inicialização de AMP
22	MPH (1 ou 2 ou ambos) identifica o elemento a ser simulado e configura a ULA
23	MPH (1 ou 2 ou ambos) recebe resultados do outro MPH e compara com os seus
24	MPH (1 ou 2 ou ambos) não converge
25	MPH (1 ou 2 ou ambos) converge e envia flag de convergência para AMP
26	MPH (1 ou 2 ou ambos) processa seus dados e envia os seus resultados para o outro MPH
27	AMP recebe flag de convergência de MPH (1 ou 2 ou ambos)
28	AMP verifica a não convergência do sistema
29	AMP verifica a convergência do sistema
30	AMP volta a esperar outro flag do arranjo
31	MPH (1 ou 2 ou ambos) recebe sinal de reinício de AMP e começa nova iteração
32	AMP pára a simulação no arranjo
33	MPH (1 ou 2 ou ambos) recebe sinal de parada de AMP e envia-lhe os resultados
34	AMP recebe os resultados do arranjo
35	AMP atualiza as variáveis de simulação e inicia nova fase
36	AMP envia os resultados da simulação para HOP e inicia a simulação doutra partição
37	AMP envia os resultados da simulação da última partição para HOP
38	HOP recebe os resultados de AMP, formata-os e envia um sinal de reinício para AMP
39	AMP conclui a simulação no arranjo
40	HOP recebe os resultados de AMP e formata-os para a saída
41	HOP envia os resultados finais, formatados, para o usuário
42	HOP conclui o processamento de saída
43	HCP finaliza a simulação

Fig. 9.7. Relação Entre as Transições do Modelo e as Ações no ABACUS.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

Cada lugar da rede modelada representa o vigorar ou não de uma dada condição ou um estado local de um processo ABACUS. Pode-se estabelecer a relação entre os lugares da figura 9.6 e os processos ABACUS, conforme a figura 9.8.

Lugares	Processo ABACUS
1-5	HCP
6-19	HIP
20-32	MPH
33-49	AMP
50-55	HOP

Fig. 9.8. Relação Entre os Lugares da RPTAN e os Processos ABACUS.

Observa-se a existência de diferentes domínios para os lugares do modelo. Adotou-se {0} para aqueles lugares onde não importa a natureza do *token*, mas sim sua presença ou ausência, caracterizando o vigorar ou não de uma condição local. Por outro lado, para os lugares componentes do processo **MPH**, foram adotados domínios envolvendo 2 tipos de *tokens*, quais sejam: o *token* **1**, representando o **MPH 1**, e o *token* **2**, representando o **MPH 2** (cada **MPH** refere-se a um dos PE's). Desta forma, tais lugares podem representar um estado local para um dos **MPH**'s ou para ambos.

Terminando a descrição da rede, deve ser notada a presença da variável **X**. Uma das condições de habilitação da transição **36** é que a transição **8** já tenha disparado, de maneira que o valor de **X** seja igual a **1**. Como os processos ABACUS são paralelos, caso não fosse utilizado esse artifício, a transição **47** poderia ser disparada sem que a transição **8** tivesse sido ainda disparada, o que não condiziria com a realidade modelada. O disparo da transição **40** restaura o valor inicial de **X**, qual seja, **0** (toda variável é igual a zero no início da simulação).

Ao simular o modelo RPTAN da arquitetura ABACUS (figura 9.6), o Simulador RPTAN gerou um grafo de classes composto por 499 classes. Além disso, concluiu que as propriedades de reiniciabilidade e de vivacidade do modelo são verdadeiras, o que garante um sistema convergente e livre de *deadlocks*.

Designando por $\mathbf{M_i}$ a marcação da classe $\mathbf{C_i}$, serão destacadas algumas das marcações alcançadas pelo processo de simulação.

Na classe inicial, tem-se:

$$\mathbf{M_0} = \{(\mathbf{1};0), (\mathbf{21};1,2), (\mathbf{28};1,2)\}$$

A marcação $\mathbf{M_0}$ significa que , no início da simulação, o lugar **1** possui o *token* **0** e os lugares **21** e **28** possuem os *tokens* **1** e **2**. Assim, a rede representa o estado inicial da arquitetura ABACUS, na qual tanto o processador hospedeiro (lugar **1**) quanto o arranjo (lugares **21** e **28**) estão à espera de um circuito para simulação.

A submissão de um circuito para simulação dispara a transição **1**, significando que o **HCP** iniciou o processo de simulação, ativando os processos **HIP**, **AMP** e **HOP**, o que é representado pela presença de *tokens* **0**, respectivamente, nos lugares **6**, **33** e **50**. Um *token* **0** é colocado também no lugar **2** (os lugares **2**, **3**, **4** e **5** representam, respectivamente, a conclusão dos trabalhos realizados pelos processos **HCP**, **HIP**, **AMP** e **HOP**). O que foi dito é representado pela marcação:

$$\mathbf{M_1} = \{(\mathbf{2};0), (\mathbf{6};0), (\mathbf{21};1,2), (\mathbf{28};1,2), (\mathbf{33};0), (\mathbf{50};0)\}$$

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

A partir desse ponto, os processos **HIP**, **AMP** e **HOP** seguem a sua execução concorrente, respectivamente processando os dados de entrada, preparando-os para o processamento no arranjo e formatando a base de dados a ser apresentada na saída. Estes processos interagem em certos pontos:

a) transições **6** e **8**;
b) transições **9**, **10** e **11**; e
c) transições **13** e **14**.

Atente para o significado do disparo dessas transições, consultando a figura 9.7.

Devido ao paralelismo da arquitetura, muitos são os caminhos possíveis dentro do grafo de classes. Exemplificando, tomando a classe $\mathbf{C_{29}}$, tem-se:

$$\mathbf{M_{29}} = \{(\mathbf{2};0), (\mathbf{11};0), (\mathbf{19};0), (\mathbf{21};1,2), (\mathbf{28};1,2), (\mathbf{36};0), (\mathbf{50};0)\}$$
$$\mathbf{D_{29}} = \{(\mathbf{8};0,0), (\mathbf{15};0,0), (\mathbf{16};0,0)\}$$

onde $\mathbf{D_{29}}$ representa o domínio de tempo da classe $\mathbf{C_{29}}$. Nesta classe, a rede encontra-se num estado onde o processo **HIP** encontra-se no estado representado pelo lugar **28**, o processo **AMP** encontra-se no lugar (estado local) **36** e o **HOP** no lugar **50**. Observa-se, pelo domínio de tempo, que, a partir da classe $\mathbf{C_{29}}$, encontram-se habilitadas as transições **8**, **15** e **16**. De acordo com o grafo de classes fornecido pelo Simulador RPTAN, a partir da classe $\mathbf{C_{29}}$, pode-se chegar às classes $\mathbf{C_{31}}$ (pelo disparo da transição **8**), $\mathbf{C_{33}}$ (pelo disparo da transição **15**) e $\mathbf{C_{34}}$ (pelo disparo da transição **16**).

Conclui-se, portanto, que o funcionamento das partes que compõem o modelo pode ser verificado a partir de várias classes, de acordo com a marcação que deseja-se analisar. É importante notar que caminhos que se bifurcam no grafo de classes, voltam a se encontrar em algum ponto, onde o sistema necessita de condições somente atendidas quando os processos estão num determinado estado. Isto pode ser constatado pela validação das propriedades de vivacidade e reiniciabilidade. Assim sendo, passa-se à descrição das principais transições do modelo da figura 9.6.

O disparo da transição **15** encerra o processamento dos dados de entrada pelo processo **HIP**. Se isto for feito na classe $\mathbf{C_{31}}$, a rede passa para a classe $\mathbf{C_{35}}$, onde a transição **16** encontra-se habilitada. Ao disparar esta última, passa-se a rede para a classe $\mathbf{C_{39}}$.

A partir da classe $\mathbf{C_{39}}$, procede-se a transferência de dados do hospedeiro para o arranjo e dá-se início à simulação do circuito nesse arranjo, o que é representado pela seqüência de disparos das transições **17**, **18** e **20**, após a qual a rede é conduzida à classe $\mathbf{C_{65}}$.

As transições **23** e **26** representam a comunicação entre os processadores do arranjo (veja as ações modeladas por essas transições na figura 9.7). A simulação continua até que o processo **AMP** decida que o arranjo convergiu para a solução do problema. Para isso acontecer, é necessária a presença dos *flags* de convergência dos dois MPH's no lugar **41**, o que ocorre na classe $\mathbf{C_{332}}$, onde:

$$\mathbf{M_{332}} = \{(\mathbf{2};0), (\mathbf{3};0), (\mathbf{30};1,2), (\mathbf{41};1,2), (\mathbf{51};0), (\mathbf{52};0)\}$$
$$\mathbf{D_{332}} = \{(\mathbf{28};0,0), (\mathbf{29};0,0)\}$$

Neste ponto (classe $\mathbf{C_{332}}$), o **AMP** decide pela convergência (transição 29) ou não (transição 28) do sistema. Caso decida pela convergência, a rede passa para a classe $\mathbf{C_{406}}$, onde:

$$\mathbf{M_{406}} = \{(\mathbf{2};0), (\mathbf{3};0), (\mathbf{30};1,2), (\mathbf{43};0), (\mathbf{51};0), (\mathbf{52};0)\}$$
$$\mathbf{D_{406}} = \{(\mathbf{32};0,0)\}$$

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

A partir da classe $\mathbf{C_{406}}$, a seqüência de disparos das transições **32** e **33** (esta, uma vez para a amarração {**1**,**2**} ou duas vezes para as amarrações {**1**} ou {**2**}, em qualquer ordem) leva à classe $\mathbf{C_{479}}$. Nesta classe, encontra-se encerrado o processamento no arranjo. Observe que os lugares **21** e **28** retornam ao seu estado inicial.

$$\mathbf{M_{479}} = \{(\mathbf{2};0), (\mathbf{3};0), (\mathbf{21};1,2), (\mathbf{28};1,2), (\mathbf{31};1,2), (\mathbf{45};0), (\mathbf{51};0), (\mathbf{52};0)\}$$
$$\mathbf{D_{479}} = \{(\mathbf{34};0,0)\}$$

Neste instante, os resultados da simulação no arranjo são transferidos para o processador hospedeiro através do disparo da transição **34**, levando a rede à classe $\mathbf{C_{483}}$:

$$\mathbf{M_{483}} = \{(\mathbf{2};0), (\mathbf{3};0), (\mathbf{21};1,2), (\mathbf{28};1,2), (\mathbf{46};0), (\mathbf{51};0), (\mathbf{52};0)\}$$
$$\mathbf{D_{483}} = \{(\mathbf{35};0,0), (\mathbf{36};0,0), (\mathbf{37};0,0)\}$$

A rede pode tomar 3 ações nesse instante, descritas a seguir:

a) Iniciar um novo instante de simulação para a mesma partição, disparando a transição **35**;
b) Transferir os resultados da simulação para o processo **HOP** e iniciar a simulação de uma nova partição, disparando a transição **36**; ou
c) Não havendo outra partição, enviar os resultados finais para o **HOP**, disparando a transição **37**.

Caso ocorra o disparo da transição **37**, chega-se à classe $\mathbf{C_{489}}$:

$$\mathbf{M_{489}} = \{(\mathbf{2};0), (\mathbf{3};0), (\mathbf{21};1,2), (\mathbf{28};1,2), (\mathbf{48};0), (\mathbf{51};0), (\mathbf{52};0), (\mathbf{53};0)\}$$
$$\mathbf{D_{489}} = \{(\mathbf{39};0,0), (\mathbf{40};0,0)\}$$

O disparo da transição **39** conclui o processo **AMP**. Disparando a transição **40**, ocorre a formatação final dos dados pelo processo **HOP**. A partir desse ponto, o disparo da transição **41** disponibiliza os dados para o usuário. O encerramento do processo **HOP** se dá pelo disparo da transição **42**, o qual leva à última classe, $\mathbf{C_{498}}$:

$$\mathbf{M_{498}} = \{(\mathbf{2};0), (\mathbf{3};0), (\mathbf{4};0), (\mathbf{5};0),(\mathbf{21};1,2), (\mathbf{28};1,2)\}$$
$$\mathbf{D_{498}} = \{(\mathbf{43};0,0)\}$$

Nesta classe, o disparo da transição **43** conduz a rede à classe inicial, concluindo com êxito a simulação.

unesp

CONCLUSÕES

O presente trabalho de mestrado originou-se de três objetivos básicos. O primeiro referia-se à aquisição, por parte do mestrando, de proficiência em assuntos relativos às Redes de Petri, as quais constituem uma ferramenta matemática de modelagem de sistemas. O segundo objetivo era o aprimoramento de uma extensão de Redes de Petri proposta em um trabalho de iniciação científica [26,27]. O terceiro e último objetivo visava a "concretização" da extensão proposta, através da implementação de um simulador computadorizado para a mesma.

Em seus 6 capítulos, a primeira parte desta dissertação trouxe uma abordagem geral sobre Redes de Petri (RP) e suas extensões mais conhecidas na literatura. Inicialmente, apresentou-se as redes em sua forma original, qual seja, a RP Elementar. Introduziu-se, assim, a composição e os conceitos básicos usados pelas principais extensões de RP. Em seguida, foi introduzida a formalização matemática de RP Lugar/Transição, o modelo de rede responsável pela alavancada do uso prático de RP. Na continuação do texto, foram apresentados dois modelos de RP de alto-nível, quais sejam, as redes Numéricas e as Coloridas. As redes de alto-nível ressaltam-se, sobretudo, por sua capacidade de trabalhar sobre *tokens* diferenciáveis e sobre conjuntos de *tokens*, tornando possível a análise do comportamento qualitativo dos sistemas por elas modelados. A primeira parte da dissertação apresentou, também, dois modelos de RP que, inicialmente propostos sobre RP Elementares e RP Lugar/Transição, podem ser estendidos para os demais modelos de RP. Tais redes são as Temporizadas e as Estocásticas. Em suas três principais versões, as redes Temporizadas adicionaram a capacidade de se trabalhar, de forma mais direta, o conceito de tempo em um modelo de sistema. Por outro lado, as redes Estocásticas adicionaram às RP a capacidade de se analisar o desempenho dos sistemas modelados, o que é feito através do cálculo de estimativas de desempenho.

A segunda parte da dissertação apresentou a proposta de uma rede de Petri concebida a partir dos principais elementos funcionais e estruturais das redes Numéricas, Temporizadas e Coloridas, a qual foi denominada Redes de Petri Temporizadas de Alto-Nível (RPTAN). Esta parte da dissertação trouxe ainda as características de implementação e funcionamento do *Simulador RPTAN*, desenvolvido para a nova extensão de RP proposta. Exemplos de modelamento em RPTAN foram apresentados com o intuito de evidenciar a praticidade e potencialidade da ferramenta.

Destarte, o desenvolvimento de RPTAN procurou aliar simplicidade, praticidade e potencialidade, resultando em uma extensão que pode ser considerada um meio-termo entre as redes Numéricas e as Coloridas, tendo a seu favor a inclusão de temporização, ausente nas redes Numéricas e há pouco tempo explorada pelas redes Coloridas. Este trabalho de desenvolvimento de RPTAN deve prosseguir através da inclusão dos conceitos de hierarquização, S-invariantes e T-invariantes, responsáveis por uma melhor análise e compreensão de uma rede, e da inclusão de componentes estruturais e funcionais que permitam a aplicação de métodos para o cálculo de estimativas de desempenho, tais como os apresentados pelas redes Estocásticas. Trata-se de um longo desafio, uma vez que podemos constatar que as redes Coloridas, as mais usadas atualmente, estão em contínuo desenvolvimento teórico, desde 1981.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

APÊNDICE: TÓPICOS DA TEORIA DE GRAFOS

A.1. DEFINIÇÕES INICIAIS

Este apêndice apresenta os aspectos da teoria de grafos mais utilizados na teoria de redes de Petri.

Um grafo é uma estrutura **G** = (**V**,**E**) onde **V** é um conjunto discreto finito e **E** é uma família cujos elementos **e** (não vazios) são definidos em função dos elementos **v** de **V**, em duas formas possíveis: se $\mathbf{E} \subset \mathbf{P}_2(\mathbf{V})$ (pares não ordenados), **G** será um grafo não orientado ou grafo simples; se $\mathbf{E} \subset \mathbf{V}^2$ (quadrado cartesiano), **G** será um grafo orientado. Veja figura A.1.

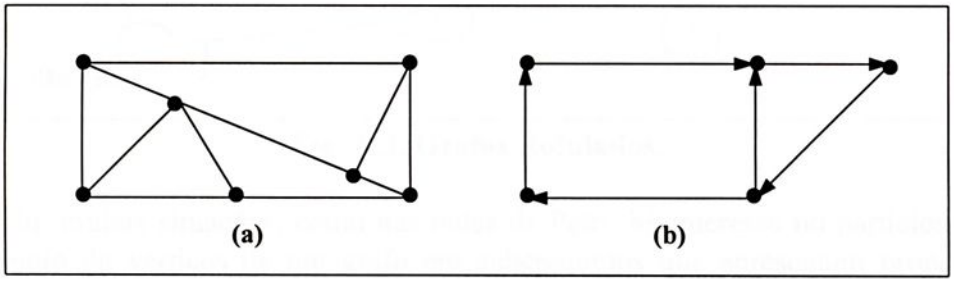


Fig. A.1. Grafos Não Orientado (a) e Orientado (b).

Os elementos de **V** são chamados **vértices**, nós ou pontos e o valor $\mathbf{n} = |\mathbf{V}|$ é a **ordem** da estrutura. A família **E** pode ser entendida como uma **relação ou conjunto de relações de adjacência** cujos elementos são chamados em geral **ligações**; em particular, nas estruturas não orientadas, os elementos **e** de **E** são conhecidos como **arestas** e, nas orientadas, como **arcos**. Dois vértices que participam de uma ligação são ditos adjacentes. O valor $\mathbf{m} = |\mathbf{E}|$ é chamado o **tamanho** do grafo. Se **m** = 0, o grafo é dito **trivial**.

Subestrutura de um grafo **G** = (**V**,**E**) é um grafo **H** = (**X**,**Y**) no qual $\mathbf{X} \subseteq \mathbf{V}$ e $\mathbf{Y} \subseteq \mathbf{E} \cap \mathbf{X}^2$. Se **X** = **V**, mas $\mathbf{Y} \subset \mathbf{E}$, a subestrutura é chamada **grafo-parcial**. Por outro lado, se $\mathbf{X} \subset \mathbf{V}$, e $\mathbf{Y} = \mathbf{E} \cap \mathbf{X}^2$, a subestrutura é chamada **subgrafo**. A figura A.2 mostra um grafo parcial e um subgrafo.

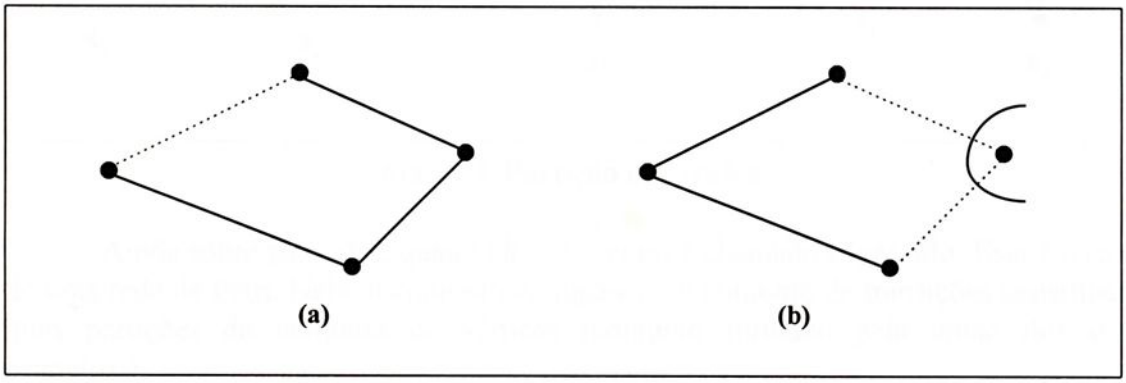


Fig. A.2. Grafo Parcial (a) e Subgrafo (b).

O **esquema** de um grafo (habitualmente chamado **o grafo**) é obtido associando-se a cada vértice um ponto ou uma pequena área delimitada por uma fronteira e a cada ligação uma linha (ou seta, no caso de grafos orientados). A caracterização de **V** como um conjunto implica na identificação dos vértices. Neste caso, fala-se de **grafos rotulados**, nos quais cada vértice é identificado por um rótulo que é, geralmente, uma palavra ou número. A identificação dos vértices por rótulos implica na identificação das ligações. Exemplos de grafos rotulados são dados na figura A.3.

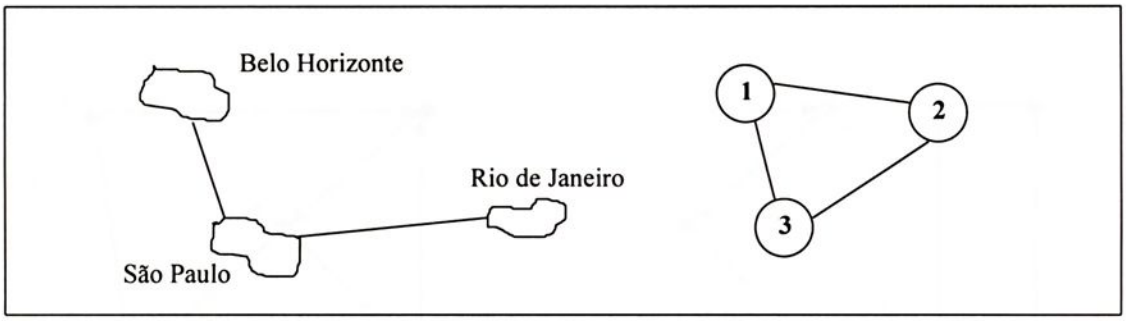


Fig. A.3. Grafos Rotulados.

Em muitas situações, como nas redes de Petri, há interesse no particionamento do conjunto de vértices de um grafo em subconjuntos que apresentam propriedades importantes para o estudo que se realiza. A mais utilizada das partições consiste em dividir o conjunto **V** em subconjuntos de vértices mutuamente não adjacentes.

Se existir uma partição **P** do conjunto de vértices de um grafo **G** = (**V**,**E**), com cardinalidade **k**, de modo que existam ligações apenas entre vértices de diferentes elementos de **P**, o grafo é dito **k-partido**. Na figura A.4, as linhas pontilhadas ilustram a separação das diferentes partições.

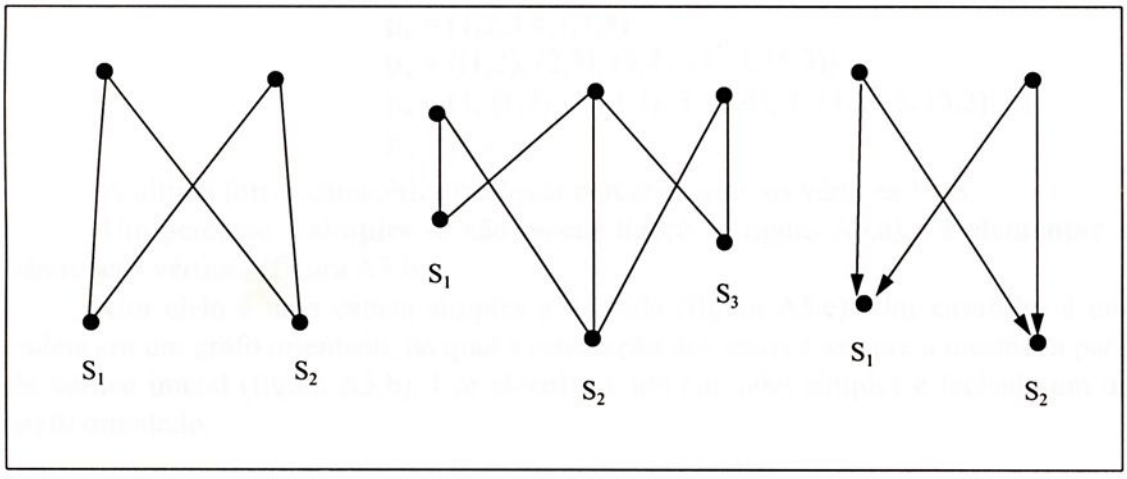


Fig. A.4. Partição de Grafos.

Ainda sobre partições, quando **k** = 2 o grafo é chamado **bipartido**. Este é o caso de uma rede de Petri. Nela, o conjunto de lugares e o conjunto de transições constituem duas partições do conjunto de vértices (conjunto formado pela união dos dois primeiros).

A.2. PERCURSOS EM UM GRAFO

Um **percurso**, ou **cadeia**, é uma família de ligações sucessivamente adjacentes. O percurso será fechado se a última ligação da sucessão for adjacente à primeira e aberto em caso contrário. Na definição geral, despreza-se implicitamente a orientação das ligações, quando se trata de grafos orientados. A figura A.5 apresenta alguns exemplos de percursos.

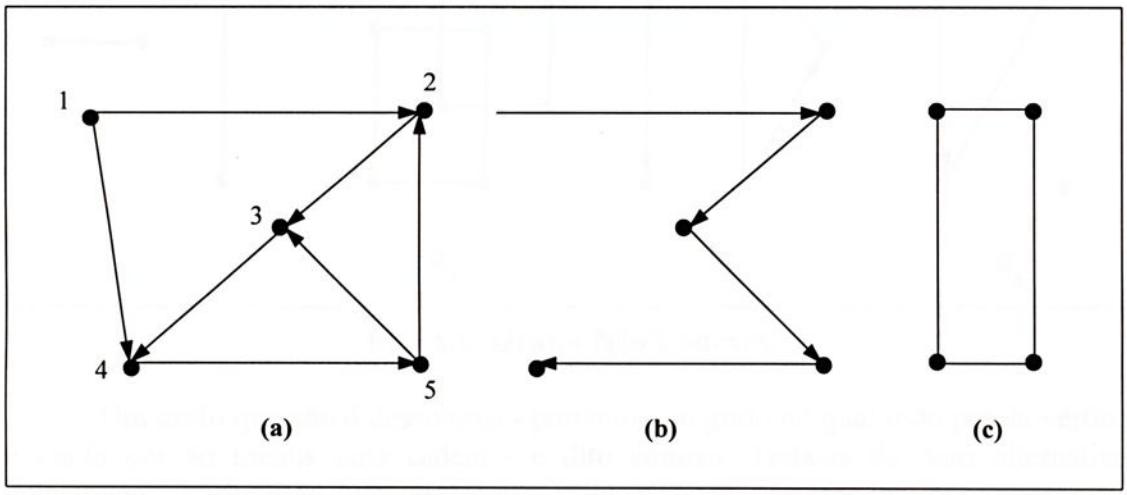


Fig. A.5. Exemplos de Percursos.

A notação é feita indicando-se a sucessão através dos vértices, das ligações ou de vértices e ligações alternados, ou apenas dos vértices inicial e final, quando isso for suficiente. Exemplificando, o percurso indicado na figura A.4.a pode ser expresso numa das seguintes formas:

$\mathbf{p_a}$ = (1,2,3,4,5,4,3)
$\mathbf{p_a}$ = ((1,2), (2,3), (3,4), (4,5), (5,3))
$\mathbf{p_a}$ = (1, (1,2), 2, (2,3), 3, (3,4), 4, (4,5), 5, (5,3), 3)
$\mathbf{p_a} = \mathbf{p}_{1\text{-}3}$

A última forma caracteriza qualquer percurso entre os vértices 1 e 3.

Um percurso é **simples** se não repetir ligações (figura A5.a) e é **elementar** se não repetir vértices (figura A5.b).

Um **ciclo** é uma cadeia simples e fechada (figura A5.c). Um **caminho** é uma cadeia em um grafo orientado, na qual a orientação dos arcos é sempre a mesma, a partir do vértice inicial (figura A5.b). Um **circuito** é um caminho simples e fechado em um grafo orientado.

A.3. CONEXIDADE EM UM GRAFO

A noção de conexidade está relacionada à **possibilidade de passagem** de um vértice a outro em um grafo através das ligações existentes. Ela traduz, portanto, o "estado da ligação" de um grafo e adquire aspectos diferentes conforme o grafo seja orientado ou não.

cm 1 2 3 4 5 6 7 unesp 11 12 13 14 15 16 17

A.3.1. TIPOS DE CONEXIDADE

Um grafo qualquer (orientado ou não) é **não conexo**, ou **desconexo**, se nele existir ao menos um par de vértices não unidos por uma cadeia. Veja figura A6.

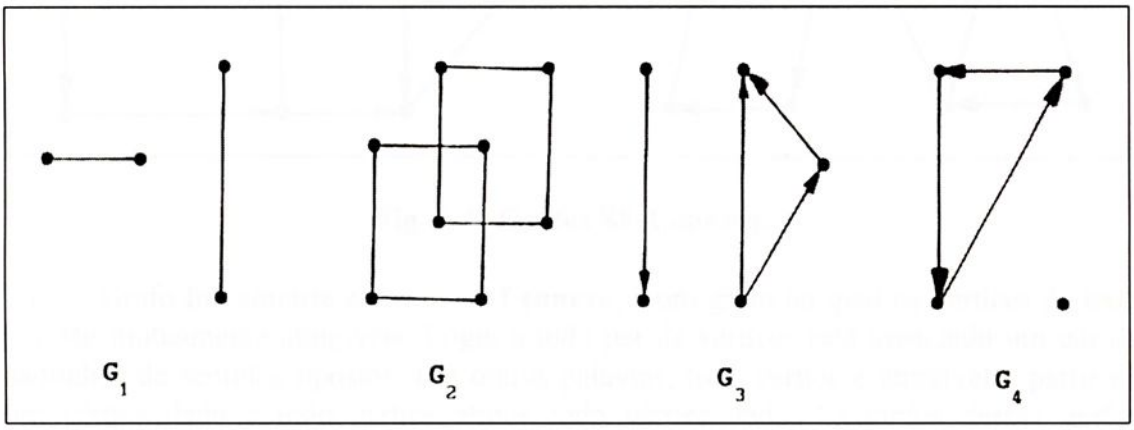


Fig. A.6. Grafos Não Conexos.

Um grafo que não é desconexo - portanto, um grafo no qual todo par de vértices é unido por ao menos uma cadeia - é dito **conexo**. Trata-se de duas alternativas mutuamente exclusivas: um grafo não conexo pode sempre ser decomposto em subgrafos conexos, que são as suas **componentes conexas**.

Já no caso de grafos orientados, alguma distinção adicional precisa ser feita.

Grafo **simplesmente conexo**, ou **s-conexo**, é um grafo no qual todo par de vértices é unido por ao menos uma cadeia. A definição é, portanto, a mesma do caso não orientado. A figura A.7 mostra alguns exemplos ilustrativos.

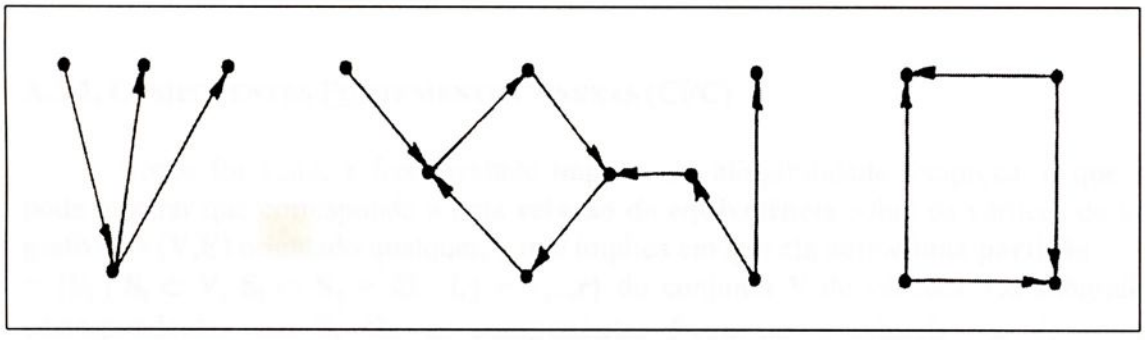


Fig. A.7. Grafos S-Conexos.

Grafo **semi-fortemente conexo**, ou **sf-conexo**, é um grafo no qual, em todo par de vértices, ao menos um é atingível a partir do outro (logo, entre eles existe um caminho em, ao menos, um dos dois sentidos possíveis). A figura A.8 apresenta alguns exemplos desses grafos.

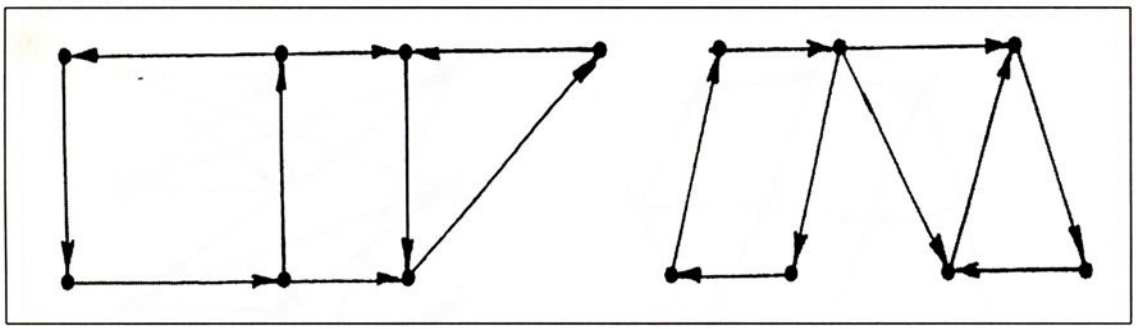


Fig. A.8. Grafos SF-Conexos.

Grafo **fortemente conexo**, ou **f-conexo**, é um grafo no qual os vértices de todo par são mutuamente atingíveis. Logo, a todo par de vértices está associado um par de caminhos de sentidos opostos. Em outras palavras, todo vértice é atingível a partir de um vértice dado e todo vértice atinge todo vértice dado. Exemplos desses grafos encontram-se na figura A.9.

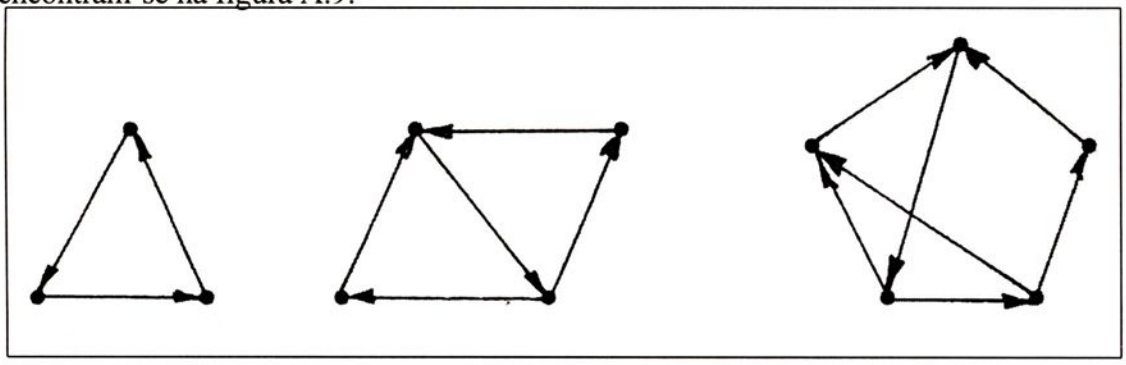


Fig. A.9. Grafos F-Conexos.

A.3.2. COMPONENTES FORTEMENTE CONEXAS (CFC)

Como foi visto, a f-conexidade implica em atingibilidade recíproca, o que se pode mostrar que corresponde a uma **relação de equivalência** sobre os vértices de um grafo $\mathbf{G} = (\mathbf{V},\mathbf{E})$ orientado qualquer, o que implica em que ela define uma **partição** $\mathbf{S} = \{\mathbf{S_i} \mid \mathbf{S_i} \subset \mathbf{V},\ \mathbf{S_i} \cap \mathbf{S_j} = \varnothing,\ \mathbf{i,j} = 1,..,\mathbf{r}\}$ do conjunto **V** de vértices. Os subgrafos correspondentes aos $\mathbf{S_i}$ são as **componentes f-conexas maximais** ou, de forma abreviada, **componentes f-conexas**. Se um grafo é f-conexo, a partição será apenas $\mathbf{S} = \{\mathbf{V}\}$.

A decomposição em componentes f-conexas pode ser observada na figura A10. O grafo à direita é o mesmo da esquerda, porém com os vértices dispostos de modo a evidenciar as componentes f-conexas correspondentes.

unesp

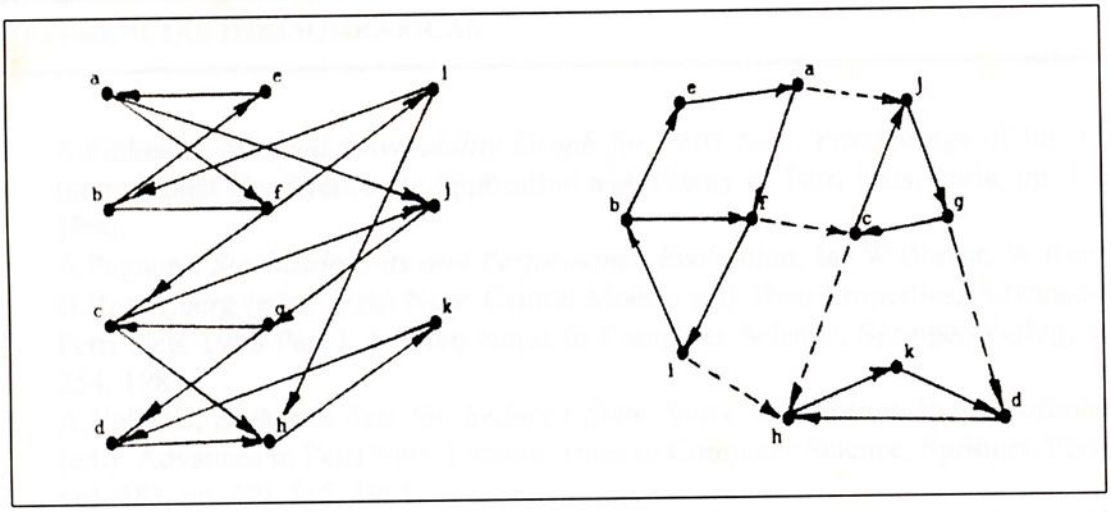


Fig. A.10. Componentes F-Conexas.

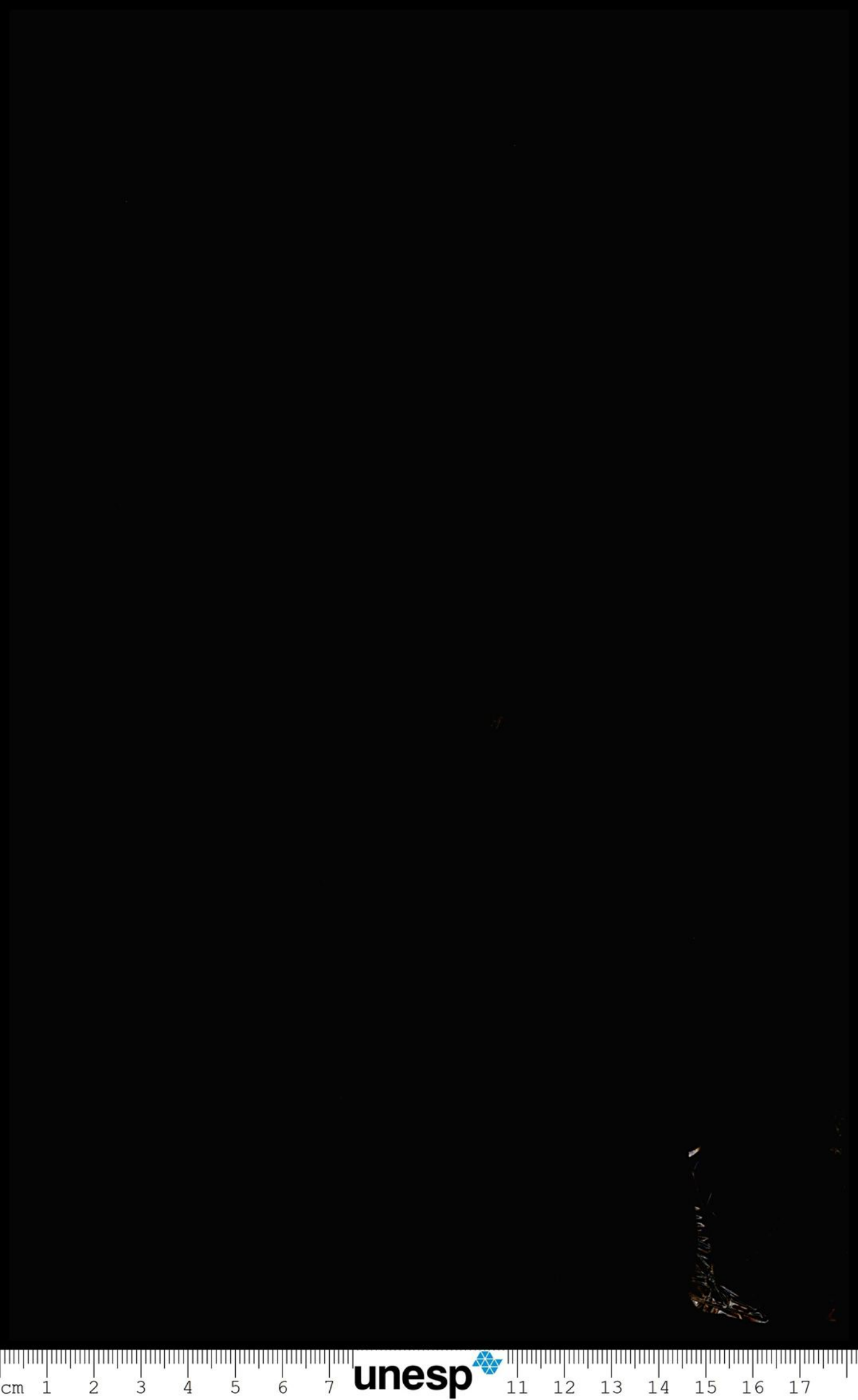
REFERÊNCIAS BIBLIOGRÁFICAS

1. A.Finkel, *A Minimal Coverability Graph for Petri Nets*, Proceedings of the 11th International Conference on Application and Theory of Petri Nets, Paris, pp. 1-21, 1990.
2. A.Pagnoni, *Stochastic Nets and Performance Evaluation*, In: W.Brauer, W.Reisig, G.Rozemberg (eds.): Petri Nets: Central Models and Their Properties, Advances in Petri Nets 1986 Part I, Lecture Notes in Computer Science, Springer-Verlag, vol. 254, 1987.
3. A.Valmari, *Stubborn Sets for Reduced State Space Generation*, In: G.Rozenberg (ed.): Advances in Petri Nets, Lecture Notes in Computer Science, Springer-Verlag, vol. 483, pp. 491-515, 1991.
4. A.Valmari, *Stubborn Sets of Coloured Petri Nets*, Proceedings of the 12th International Conference on Application and Theory of Petri Nets, Aarhus, pp. 102-121, 1991.
5. B.Berthomieu, M.Diaz, *Modeling and Verifications of Time Dependent Systems Using Time Petri Nets*, IEEE Transactions on Software Engineering, 17 (3) 259-273, March 1991.
6. B.C.Damasceno, A.Yamakami, *FMS: A Tool for Analysis and Simulation Based on Numerical Petri Nets (NPN)*, TIMS XXX - SOBRAPO XXIII, 1991.
7. B.C.Damasceno, M.Marton, N.Marranghello, F.Damiani, *Um Modelo em Rede de Petri de uma Arquitetura para a Simulação de Circuitos*, revista Controle & Automação, 6 (2) 57-63, 1995.
8. C.Ghezzi, D.Mandrioli, M.Pezzè, S.Morasca, *A Unified High-Level Petri Net Formalism for Time-Critical Systems*, IEEE Transactions on Software Engineering, 17 (2) 166-167, February 1991.
9. C.Heuser, *Modelagem Conceitual de Sistemas: Redes de Petri*, V EBAI, 1990.
10. F.J.W.Symons, *Introduction to Numerical Petri Nets, a General Graphical Model of Concurrent Processing Systems*, Australian Telecommunications Research, 14 (1) 28-33, 1980.
11. G.Florin, S.Natkin, *Evaluation Based upon Stochastic Petri Nets of the Maximum Troughput of a Full Duplex Protocol*, C.Girault, W.Reisig, Informatik-Fachberichte, Springer-Verlag, vol. 52, 1982.
12. H.J.Genrich, K.Lautenbach, *System Modelling with High-level Petri Nets*, Theoretical Computer Science, North-Holland, vol. 13, pp. 109-136, 1981.
13. J.B.Dugan, *Extended Stochastic Petri Nets: Aplications and Analysis*, Ph. D. Dissertation, Department of Eletrical Engeneering, Duke University, July 1984.
14. J.L.Peterson, *Petri Net Theory and Modelling of Systems*, Prentice-Hall, 1981.
15. K.Jensen, *Coloured Petri Nets and the Invariant Method*, Theoretical Computer Science, North-Holland, vol. 14, pp. 317-336, 1981.
16. K.Jensen, *High-level Petri Nets*, In: A.Pagnoni, G.Rozenberg (eds.): Applications and Theory of Petri Nets, Informatik-Fachberichte, Springer-Verlag, vol. 66, pp. 166-180, 1983.
17. K.Jensen, G.Rozenberg (eds.): *High-level Petri Nets. Theory and Application*, Springer-Verlag, 1991.

unesp

18. K.Jensen, *Coloured Petri Nets. Basic Concepts, Analysis Methods and Pratical Use*, EATCS Monographs on Theoretical Computer Science, Springer-Verlag, vol. 1, 1992.
19. K.Jensen, *An Introduction to the Theoretical Aspects of Coloured Petri Nets*, In: J.W. de Bakker, W.-P. de Roever, G.Rozenberg (eds.): A Decade of Concurrency, Lecture Notes in Computer Science, Springer-Verlag, vol. 803, pp. 230-272, 1994.
20. L.Petrucci, *Combining Finkel's and Jensen's Reduction Techniques to Build Covering Trees for Coloured Nets*, Petri Nets Newsletter, Special Interest Group on Petri Nets and Related System Models, Gesellschaft für Informatik (GI), Germany, vol. 36, pp. 32-36, August 1990.
21. M.A.Marsan, G.Balbo, G.Conte, *A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multi-Processors Systems*, ACM Trans. on Computer Systems, 2 (2), May 1984.
22. M.Marton, W.C.Borelli, S.Motoyama, *Rede de Petri Temporizada: Modelo e Exemplo*, Relatório Técnico 077/86, FEE/UNICAMP, 1986.
23. M.Moloy, *On the Integration of Delay and Throughput Measures in Distributed Processing Models*, UCLA Computer Science Department Report N.CSD-810921, University of California, Los Angeles, 1981.
24. N.Marranghello, F.Damiani, *ABACUS: A Hardware-Based Circuit Simulator*, SID Digest of Technical Papers, pp. 177-179, 1990.
25. R.T.Tamura, S.Motoyama, W.C.Borelli, *Rede de Petri: Análise e Simulação*, Relatório Técnico 029/85, FEE/UNICAMP, 1985.
26. W.L.A.Oliveira, N.Marranghello, *Estudo Inicial: Redes de Petri e Arquiteturas de Computadores*, Relatório I - FAPESP - Processo 93/4986-2, Agosto 1994.
27. W.L.A.Oliveira, N.Marranghello, *Rede de Petri Numérica no Modelamento de uma Arquitetura: Simulador RPNT e Arquitetura ABACUS*, Relatório II - FAPESP - Processo 93/4986-2, Fevereiro 1995.
28. W.Reisig, *Petri Nets: an Introduction*, Springer, Berlin-Heidelberg, 1985.

unesp


cm 1 2 3 4 5 6 7 11 12 13 14 15 16 17
unesp