UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UM SIMULADOR DE REDES DE PETRI DE ALTO NÍVEL PARA USO DIDÁTICO

por

FLÁVIO SOIBELMANN GLOCK

Dissertação submetida como requisito parcial para
a obtenção do grau de Mestre em
Ciência da Computação

Prof. Carlos Heuser

Orientador

Porto Alegre, agosto de 1992

CIP - CATALOGAÇÃO NA PUBLICAÇÃO

Glock, Flávio Soibelmann
Um simulador de redes de Petri de alto nível para uso didático / Flávio Soibelmann Glock. – Porto Alegre : CPGCC da UFRGS, 1992.
111p. : il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul, Curso de Pós-Graduação em Ciência da Computação, Porto Alegre, 1992. Orientador: Heuser, Carlos

Dissertação: Redes de Petri, Ensino, Simulador

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

O presente trabalho consiste na apresentação da implementação de um software projetado para auxiliar o ensino de redes de Petri de alto nível. A partir da fundamentação sobre o assunto desenvolvida em cursos introdutórios de modelagem de sistemas com redes de Petri foi especificado um sistema para ser usado em sala de aula.

O sistema desenvolvido permite a demonstração das características gráficas e da semântica de alguns modelos tradicionais de redes de Petri, tais como redes condição/evento, redes predicado-transição e redes coloridas. Redes de baixo nível também podem ser demonstradas.

O software foi elaborado de forma a permitir a utilização de redes já definidas em disquete. O usuário pode também definir novas redes utilizando o editor gráfico/compilador ou se valer de uma linguagem textual de baixo nível para descrever as redes. Após a obtenção da rede desejada, é possível a realização de demonstrações, simulando o funcionamento da mesma.

O software permite ainda a execução de passos, o funcionamento "para trás" da rede e identifica, a cada estado alcançado, quais alterações estão habilitadas e quais estão em conflito. Um comando permite obter o caminho entre duas marcações da rede. O usuário pode selecionar regiões de interesse sobre a rede, que serão mostradas com mais detalhe na tela.

Usando o editor gráfico/compilador o usuário pode modificar redes durante o decorrer da apresentação.

A interface com o usuário, voltada para o uso em sala de aula, necessita como hardware um projetor de vídeo apropriado e um computador pessoal equipado com mouse. A utilização do compilador em sala de aula exige, no entanto, mais recursos computacionais que o simulador.

Alguns modelos de redes de Petri não são suportados pelo compilador, mas podem ser implementados utilizando a linguagem de baixo nível do sistema.

O usuário, professor ou aluno, encontrará neste trabalho instruções para o emprego prático do sistema como instrumento para a realização de demonstrações didáticas.

PALAVRAS-CHAVE: Redes de Petri, Ensino, Simulador

TITLE: "A petri-net simulator for didactic use"

ABSTRACT

This work describes the implementation of a software designed as a teaching aid for instructors of high-level Petri nets.

A system for class-room use is described that is based on material developed during introductory courses of systems modelling with Petri nets.

The class-room system permits the demonstration of the graphic properties and the semantics of some traditional nets, such as condition/event nets, predicate-transition nets and coloured nets, as well as low level nets.

The software was designed to permit the use of nets already defined in diskettes. The user can define new nets using either the graphic editor/compiler or a low level textual language.

Some models of Petri nets are not supported by the compiler, but can be implemented using the low level language.

After obtaining the desired net, it is possible to perform demonstrations, simulating its operation.

The software also allows step by step execution of nets, including backwards execution, and identifies, for every state, which alterations may happen and which are in conflict.

A command permits the user to obtain the path between two markings of the net. The user may select regions of interest over the net, that will be shown with more detail in the screen.

Using the graphic editor/compiler the user may modify the nets during the presentation.

The hardware needs are a video projector and a personal computer equiped with a mouse.

KEYWORDS: Petri-nets, teach, simulator

1 INTRODUÇÃO

1.1 Considerações gerais

A elaboração e análise de estruturas organizacionais tem encontrado um instrumento de grande funcionalidade a partir de 1961 quando C. A. Petri /HEU 90/ propôs o uso de redes que receberam o seu nome para a modelagem de sistemas onde aparecem componentes concorrentes.

As redes de Petri representam um esforço continuado para desenvolver conceitos, teorias e ferramentas para auxiliar no projeto e análise de sistemas concorrentes.

A utilização das redes de Petri como metodologia depende em parte da sistematização do conhecimento já existente e de sua divulgação para a comunidade de usuários que pertencem às áreas de engenharia de software, banco de dados e sistemas de informação, arquitetura de computadores e sistemas operacionais, protocolos de comunicação e redes de computadores, controle de processos e automação de escritórios /BRA 87/.

Os pesquisadores da área de redes de Petri vêm reconhecendo ao longo dos anos a necessidade de ferramentas para o ensino que permitam apresentar a metodologia, descrevê-la, demonstrar seu uso, para depois discutir sua utilidade no auxílio ao desenvolvimento de projetos /CIN 86/.

A ênfase dada à produção de sistemas comerciais de suporte a redes de Petri produziu sistemas que também atenderiam às necessidades de ensino. A necessidade de adequar estes sistemas às necessidades práticas reais produziu, no entanto, sistemas excessivamente exigentes de capacidade de processamento, para suportar grandes modelos gráficos executáveis /ALB 89/.

As experiências com sistemas voltados exclusivamente para ensino têm sido limitadas a redes elementares ou outros modelos simplificados, e suportam principalmente a análise matemática das redes /JEN 84/ /MAZ 90/.

Para especificar um sistema a ser usado em sala de aula, partiu-se do levantamento das redes que são apresentadas em um curso introdutório. Estas redes são representadas pelos exemplos dados em um livro texto /HEU 90/, utilizado nos cursos de Modelagem de Sistemas com Redes de Petri da Escola Brasileiro-Argentina de Informática (EBAI), e no Curso de Pós-graduação em Ciências da Computação da Universidade Federal do Rio Grande do Sul.

Tais redes caracterizam-se por seu pequeno tamanho e pela linguagem de anotação complexa (redes de alto nível).

Neste trabalho optou-se por partir da necessidade do professor, representada pela existência de pequenas redes de alto nível que seriam demonstradas em sala de aula, para especificar e implementar um software capaz de apresentar estas redes em um sistema gráfico de pequena capacidade de processamento, que é freqüentemente disponível nos cursos de computação. Este sistema mínimo seria composto por um microcomputador do tipo PC-XT, um mouse e uma tela de cristal líquido para retro-projeção.

1.2 Abordagem informal

Uma abordagem informal para introduzir de maneira ampla o que seja *rede* pode ser atingida a partir da associação de três componentes: O primeiro será chamado *conjunto de lugares* S onde cada lugar representa uma *condição* para que um determinado evento aconteça. Na figura 1 a condição é simbolizada pela *elipse* s "carro na garagem". Exemplo: Para que um carro possa sair de uma garagem é imperativa a condição de que ele lá esteja.

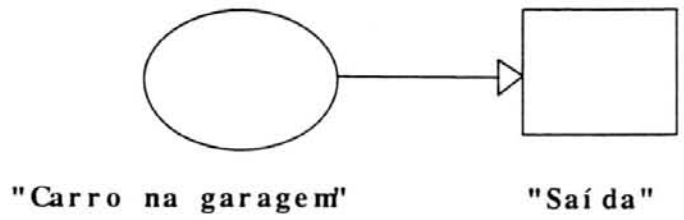


Figura 1.1 Exemplo de rede

O segundo componente será chamado *conjunto de conexões* T onde cada conexão representa um evento. No exemplo em foco a conexão consistirá no único evento possível que é a "saída" do carro, uma vez que foi estabelecida a condição de que ele está no interior da garagem. Na figura 1 a conexão é representada pelo *retângulo* t "saída".

O terceiro componente é o *conjunto de relações* F onde cada relação representa a obrigatoriedade ou a proibição de uma condição para que um evento ocorra. No exemplo citado a conexão "saída" somente será habilitada se a condição "carro na garagem" for satisfeita. Na figura 1 a relação é simbolizada pela *seta* f que aponta da elipse para o retângulo.

Por convenção a *obrigatoriedade* da existência de determinada condição para que um evento ocorra é simbolizada pelo par ordenado <lugar, conexão> ou <condição, evento> ou ainda, como na figura 1, por uma seta que parte do lugar para a conexão.

De maneira análoga, a *proibição* da existência de determinada condição para a ocorrência do evento é simbolizada pelo par ordenado de forma oposta: <conexão, lugar> ou, na representação gráfica, por uma seta que parte da conexão para o lugar.

1.3 Abordagem formal

A rede descrita na abordagem informal acima pode ser representada matematicamente como

$$N = < \{ \text{"carro na garagem"} \}, \{ \text{"saída"} \}, \{ < \text{"carro na garagem"}, \text{"saída"} > \} >$$

Generalizando, pode-se enunciar

Definição: Uma *rede* N é uma tripla

$$N = < S, T, F >$$

onde

(1) S é um conjunto de lugares

(2) T é um conjunto de conexões

(3) $S \cap T = \emptyset$ e $S \cup T \neq \emptyset$

(4) $F \subseteq (S \times T) \cup (T \times S)$

sendo que, para satisfazer (3) e (4), N, S, T e F não podem ser vazios, e S e T devem ser disjuntos.

1.3.1 Redes de Petri

A teoria de redes é para Genrich /GEN 87/ uma *teoria de sistemas* que busca a compreensão de sistemas cuja estrutura e comportamento são determinados pela natureza distributiva e combinatória de seus estados e alterações.

Os *sistemas condição/evento* foram propostos por Petri /PET 76/ como um modelo de referência para a teoria de redes.

Todo o modelo que pode ser derivado deste modelo de referência, ou transformado para ele, é considerado, em um sentido estrito, um *modelo teórico de rede*.

Se um sistema dinâmico tem uma representação adequada em um modelo teórico de rede, ele é chamado de sistema de Petri ou *rede de Petri*.

1.3.1.1 Rede condição/evento

Se o modelo matemático acima for restrito a redes simples e sem lugares ou conexões isolados e for acrescentado um quarto elemento C conhecido como *classe de alcançabilidade total* constituir-se-á uma rede condição/evento, permitindo-se enunciar

Definição: Uma rede *condição/evento* Σ é uma quádrupla

$$\Sigma = < S, T, F, C >$$

onde

(1) $< S, T, F >$ é uma rede *simples* e sem lugares ou conexões isolados

(1.1) Uma rede é *simples* se, para $x, y \in (S \cup T)$ tivermos

$$({}^{\bullet}x = {}^{\bullet}y \wedge x^{\bullet} = y^{\bullet}) \Rightarrow (x = y)$$

sendo que o *pré-conjunto de x* $\underline{{}^{\bullet}x}$ é

$${}^{\bullet}x = \{ y \mid <y, x> \in F \}$$

e o *pós-conjunto de x* $\underline{x^{\bullet}}$ é

$${}^{\bullet}x = \{ y \mid <x, y> \in F \}$$

(1.2) Uma rede não tem lugares ou conexões isolados se, para $s \in S, t \in T, f \in F$ tivermos

$$\forall s \; \exists f \; (f = <s, t> \vee f = <t, s>) \wedge$$
$$\forall t \; \exists f \; (f = <s, t> \vee f = <t, s>)$$

(2) A classe de alcançabilidade total C é o subconjunto

$$C \subseteq \text{POTENCIA} (S)$$

representando todas as marcações da rede que precedam outra marcação ou que sejam sucessoras de outra marcação.

(2.1) Uma *marcação* é um subconjunto $\underline{c}$ tal que

$$c \subseteq S$$

(2.2) Uma marcação $\underline{c'}$ é *sucessora* de $\underline{c}$ sob uma conexão t, para $t \in T$, $c \subseteq S$, $c' \subseteq S$, escrevendo-se

$$c [t > c'$$

onde

$$c' = (c \setminus \bullet t) \cup t\bullet$$

se t está *habilitado* em c, ou seja

$$\bullet t \subseteq c \ \wedge \ t\bullet \subseteq S \setminus c$$

(2.2.1) A marcação c é dita *predecessora* de c'.

1.3.2 Ramos restauradores

Dada uma rede condição/evento

$$\Sigma = < S, T, F, C >$$

para

$$f, f_1, f_2 \in F, \ t \in T, \ s \in S, \ c \in C, s_1 \in c,$$

se

$$f_1 = <s, t>, \ f_2 = <t, s>$$

a conexão t nunca estará habilitada, pois

$$\bullet f \subseteq c \ \wedge \ f\bullet \subseteq S \setminus c$$

é sempre falso, já que um lugar s_1 não pode estar contido em c e $S \setminus c$ ao mesmo tempo, o que será discutido a seguir.

Na modelagem de um semáforo, por exemplo, uma condição não poderia participar em uma alteração e ainda estar em vigor na marcação sucessora (a fim de manter o estado do semáforo).

Para resolver este problema, é associado a cada par f ("ramo") um atributo de "alterador" ou de "restaurador". A definição de *marcação sucessora* é então adaptada de forma que um *ramo restaurador* não possa alterar uma condição.

1.3.3 Conexões vivas e conexões mortas

Dada uma conexão *t* pertencente a uma rede condição/evento, *t* está habilitado para uma determinada classe de condições em vigor na rede.

Atribuindo a *t* um atributo de "viva" ou "morta" podemos definir que: Se *t* é viva, valem as regras de habilitação já definidas. No entanto, se uma marcação sucessora sob a conexão viva *t* habilitar uma *conexão morta*, aquela marcação sucessora não é *alcançável.*

1.3.4 Redes de Petri compactas

1.3.4.1 Definição

Segundo Heuser /HEU 90/, uma rede compacta é composta por

(1) Uma *linguagem de anotação* (LA)

(2) Uma definição de um conjunto de *universos de discurso* (UD)

(3) Uma *rede de Petri* anotada com a LA como se segue:

(3.1) A cada conexão é associada uma fórmula da LA. Esta fórmula, a *fórmula de conexão*, aparece, na representação gráfica da rede, dentro do retângulo representativo da conexão. No caso especial da fórmula ser *Verdadeiro*, ela pode ser omitida.

(3.2) A cada lugar são associados dois termos constantes (termos sem variáveis livres) da LA, que devem designar, cada qual, um conjunto do UD. Um dos termos, o *domínio do lugar*, aparece, na representação gráfica da rede (figura 1.2)entre parênteses, após o identificador do lugar. O outro termo, o designador da *marcação inicial*, aparece, na representação gráfica, dentro da elipse representativa do lugar. No caso especial de a marcação inicial ser o conjunto vazio, o termo pode ser omitido.

(3.3) A cada ramo é associado um *termo*. Este termo, o *termo de ramo*, deve designar um conjunto de entidades do UD, e aparece, na representação gráfica da rede, junto à seta representativa do ramo.

1.3.4.2 Universo de discurso

Quando criamos um *modelo* de sistema, identificamos, na realidade modelada, objetos que queremos representar em nosso modelo.

A estes objetos damos a denominação de *entidades*. O conjunto de todas as entidades que consideramos em um modelo forma o *universo de discurso* deste modelo.

1.3.4.3 Linguagem de anotação

A *linguagem de anotação* para redes compactas é uma linguagem de lógica de predicados de primeira ordem.

a) Semântica

A *linguagem de anotação* serve para escrever expressões acerca das entidades que compõem o universo de discurso. Estas expressões são de dois tipos:

- *termos*, que designam, cada um, uma entidade do universo de discurso

- *fórmulas*, que designam os valores lógicos *falso* ou *verdadeiro*

b) Sintaxe

A sintaxe da linguagem de anotação é apresentada na tabela 1.1 em sua forma geral. Símbolos de relação e símbolos de função devem ser definidos para cada modelo.

Heuser /HEU 90/ apresenta uma definição detalhada da sintaxe e semântica da linguagem de anotação.

Tabela 1 Sintaxe da linguagem de anotação

<fórmula> ::=	*verdadeiro* | *falso* | <símb. relação> (<termo>, ...) | (<termo> <símb. relação binária> <termo>) | (<fórmula> <símb. lógico> <fórmula>) | *não* <fórmula> | paratodo <var> (<fórmula>) | existe <var> (<fórmula>) | (<fórmula>)
<símb. lógico> ::=	*e* | *ou* | *impl*
<símb. relação> ::=	(de acordo com o modelo)
<símb. relação binária> ::=	*elem* | *sub* | = | (outros de acordo com o modelo)
<termo> ::=	<constante> | <var> | <símb. função> (<termo>, ...) | (<termo> <símb. função binária> <termo>) | { <termo>, ... } { <var> | <fórmula> } | < <termo>, ... > | (<termo>)
<símb. função> ::=	*POTEN* | (outros de acordo com o modelo)
<símb. função binária> ::=	× | (outros de acordo com o modelo)

1.3.4.4 Funcionamento da rede compacta

O funcionamento da rede dá-se através do aparecimento e desaparecimento de entidades de lugares da rede compacta.

Uma marcação na rede compacta é dada pela presença ou ausência de entidades nos lugares.

As entidades que podem aparecer em um lugar são definidas pelo domínio do lugar.

Cada conexão define um conjunto de alterações.

As alterações definidas por uma conexão são indicadas pelos termos dos ramos, pela fórmula de conexão e pelos domínios dos lugares conectados.

1.3.5 Definições aplicáveis às redes compactas

a) Marcas definidas por um lugar

Seja uma rede compacta e um UD definido para esta rede. Um lugar da rede compacta define uma marca para cada entidade do domínio do lugar. É usual identificar a marca através do par

(*identificador do lugar da rede compacta , entidade*)

b) Marcação

Um conjunto qualquer de marcas definidas pelos lugares da rede forma uma *marcação da rede.*

Um conjunto qualquer de marcas definidas por um lugar forma a *marcação do lugar.*

c) Alterações definidas por uma conexão

Seja uma rede compacta e um UD definido para esta rede. Uma conexão da rede define uma alteração para cada valoração de suas variáveis que obedeça às seguintes condições:

- O conjunto de entidades designado pelo termo de cada ramo, sob a valoração em questão, deve fazer parte do domínio do lugar conectado.

- A fórmula de conexão, caso presente, deve resultar, sob a valoração em questão, em verdadeiro.

A *conexão subjacente*, representando a conexão na rede condição/evento equivalente a esta rede compacta (*rede subjacente*), é representada por

identificador da conexão compacta; valores das variáveis

Os valores das variáveis de conexão são dados em ordem alfabética de nome de variável.

d) Marcas ligadas a uma alteração

As marcas ligadas a uma alteração são as marcas correspondentes às entidades designadas pelos ramos da conexão sob a valoração que define a alteração.

Estas marcas podem ser classificadas em:

- marcas de entrada: marcas correspondentes às entidades designadas pelos termos dos ramos de entrada da conexão

- marcas de saída: marcas correspondentes às entidades designadas pelos termos dos ramos de saída da conexão

e em:

- marcas alteradas: marcas correspondentes às entidades designadas pelos termos dos ramos alteradores da conexão

- marcas restauradas: marcas correspondentes às entidades designadas pelos termos dos ramos restauradores da conexão.

Podem ser feitas combinações destes dois sistemas de classificação das marcas de uma alteração, existindo então marcas restauradas de saída, restauradas de entrada, alteradas de saída e alteradas de entrada.

e) Regra de habilitação

Uma alteração está habilitada frente a uma marcação quando

- suas marcas de entrada, caso a alteração as tenha, estão presentes na marcação

- suas marcas de saída, caso a alteração as tenha, estão ausentes na marcação.

f) Concorrência

Diz-se que duas alterações habilitadas são concorrentes quando elas não possuem marcas de entrada comuns, nem marcas de saída comuns.

g) Conflito

Diz-se que duas alterações estão em conflito quando possuem ao menos um lugar de entrada comum, ou ao menos um lugar de saída comum.

h) Passo

Um conjunto de alterações é um passo frente a uma marcação, chamada marcação precursora, quando

- todas alterações do passo estão habilitadas dentro da marcação precursora

- as alterações do passo não são conflitantes entre si.

i) Efeito da ocorrência de um passo

O efeito da ocorrência das alterações de um passo é a transição da marcação precursora para uma marcação sucessora de tal forma que

- todas as marcas alteradas de entrada das alterações do passo desapareçam da marcação sucessora

- todas as marcas alteradas de saída das alterações do passo apareçam na marcação sucessora.

j) Marcação alcançável

Frente a uma marcação inicial da rede (marcação qualquer da rede) define-se marcação alcançável como sendo qualquer marcação obtida pelo efeito da ocorrência de um passo sobre a marcação inicial ou sobre marcações alcançáveis a partir desta.

1.3.5.1 Conclusão

As redes compactas são, pois, uma classe de redes de Petri onde cada símbolo pode representar mais de um elemento do mundo real. Estas redes são também conhecidas como "redes alto-nível" na literatura.

As redes compactas utilizam notação matemática derivada da teoria de conjuntos.

As redes compactas podem ser utilizadas em um computador, bastando que sua linguagem seja "implementada" na máquina.

As redes compactas necessitam de uma "interpretação" para tornarem-se modelos úteis na prática. A figura 2 mostra uma rede compacta correta, mas sem utilidade como modelo conceitual.

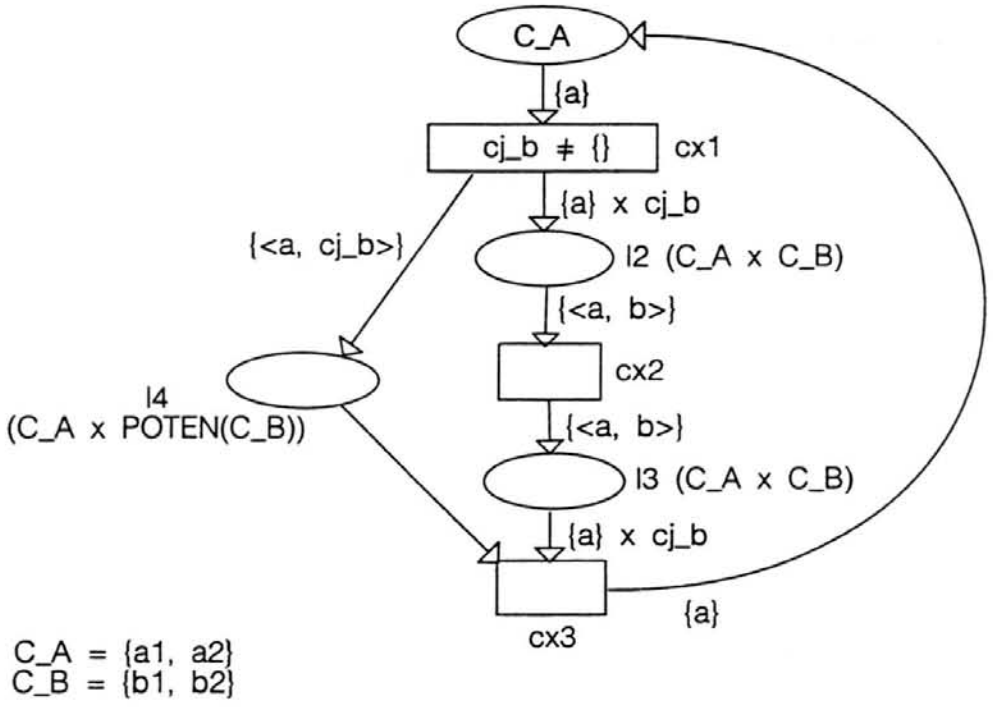


Figura 1.2 Exemplo de rede compacta

1.4 Redes predicado/transição e redes coloridas

As redes predicado/transição /GEN 87/ e as redes coloridas /JEN 87/ deram origem às redes compactas descritas anteriormente.

As principais diferenças destas redes são:

- em alguns pontos da linguagem de anotação
- no tratamento de multiconjuntos (ou "bags")
- as redes predicado/transição não têm conjuntos nos ramos
- redes coloridas consideram apenas os lugares de entrada na regra de habilitação.

1.5 Sistemas condição/evento e redes elementares

As redes elementares são um tipo de sistema condição/evento que distingue-se principalmente pela terminologia utilizada, resumida na tabela 2.

Os sistemas condição/evento e as redes elementares são considerados neste trabalho como um caso especial das redes compactas, onde não é permitida a utilização de conjuntos e onde o universo de discurso contém apenas um elemento, denominado *marca elementar*. A marca elementar é representada por um ponto preto.

As redes elementares são utilizadas no curso básico de redes de Petri para introduzir os conceitos de "funcionamento" das redes.

A figura 3 apresenta uma rede elementar que constitui-se em um modelo dos estados e transições dos sinais durante a comunicação entre um modem *m* e um computador *c*.

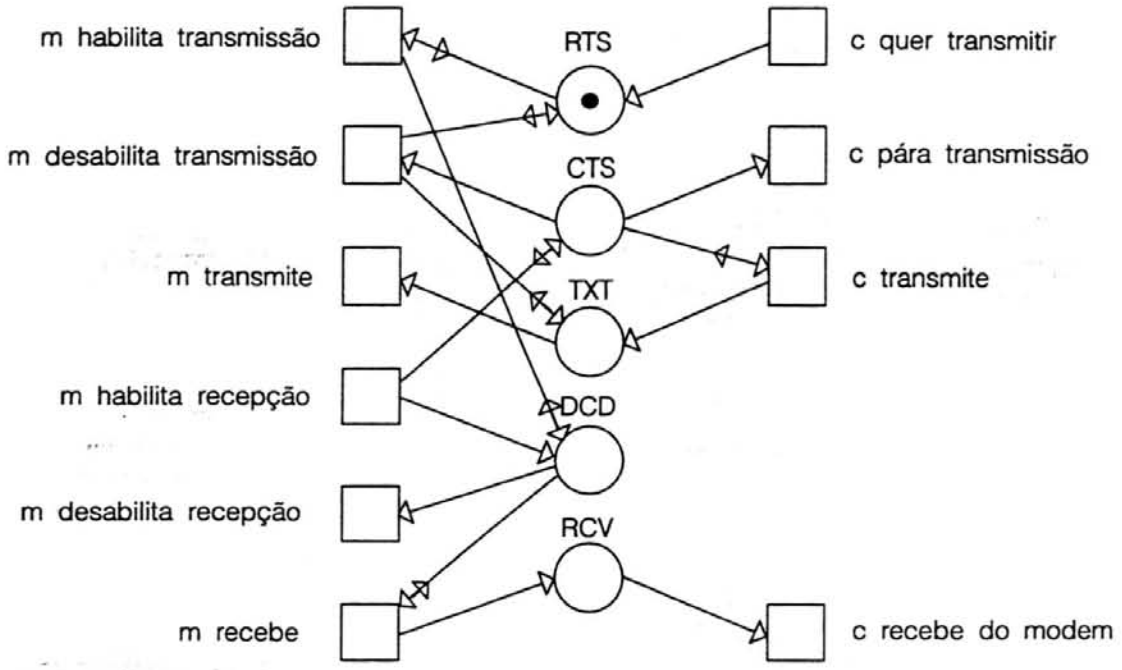


Figura 1.3 Exemplo de rede elementar

1.6 Redes lugar/transição

As redes lugar/transição (figura 4) diferenciam-se das redes elementares por apresentar uma *capacidade* dos lugares e pela terminologia utilizada, resumida na tabela 2.

Tabela 1.2 Diferença de terminologia entre diversas redes

Sistema condição/evento	Rede elementar	Rede lugar/transição
marca	condição	ficha
presença de marca	vigorar de condição	possuir fichas
alteração	evento	transição
marcação da rede	estado do modelo	marcação da rede

A capacidade de um lugar pode ser um número natural ou ilimitada (simbolizada por ω). Cada ramo da rede tem associado a ele um número natural, especificando quantas fichas aparecem ou desaparecem do lugar por efeito da ocorrência da transição. Uma marcação da rede passa a ser a associação, a cada um dos lugares da rede, de uma determinada quantidade de fichas, respeitando a capacidade de cada um.

As redes lugar/transição têm como campo principal de aplicação a análise *quantitativa* de sistemas. Na modelagem conceitual de sistemas, por outro lado, estamos mais interessados em representar as *relações causais* que aparecem na realidade modelada (conflito, concorrência, seqüência) para obter um retrato fiel desta realidade.

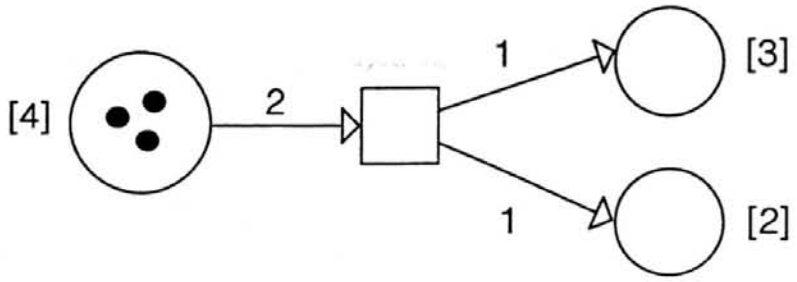


Figura 1.4 Exemplo de rede lugar/transição

2 OBJETIVOS

2.1 Objetivo geral

O objetivo do sistema desenvolvido é servir de ferramenta para o ensino de modelagem de sistemas com redes de Petri em sala de aula. O operador do sistema é o professor, que utiliza o computador dentro de um sistema de projeção que possibilita a visualização por todos os alunos presentes. O objetivo do sistema é acompanhar a explanação do professor, diminuindo a utilização do quadro negro.

2.2 Objetivos específicos

2.2.1 A quem se destina

O sistema destina-se a três classes de usuários:

O modelador fará a descrição das redes a serem utilizadas em sala de aula. Esta descrição ficará armazenada em disco flexível para utilização posterior. O processo de descrição de rede não é crítico para o funcionamento do sistema, pois será feito fora da sala de aula e apenas uma vez para determinado tipo de curso.

O professor utilizará o sistema para expor graficamente conceitos sobre redes. O sistema responderá ao professor utilizando uma interface simples de compreender e contextual, de forma que um mínimo de comandos básicos sejam necessários. O professor completará sua exposição utilizando outros meios, tais como o quadro-negro, para complementar os conceitos teóricos não transmitidos ao aluno pelo sistema.

O aluno receberá do sistema uma imagem gráfica da matéria que está sendo exposta pelo professor. Esta imagem normalmente estará projetada em uma tela à sua frente ou, caso isto não seja possível, estará em um monitor de vídeo colocado para um número pequeno de alunos.

2.2.2 Características necessárias do sistema

O sistema deve poder ser executado sobre um sistema mínimo, popular nos cursos de computação e que possa ser instalado em uma sala de aula. Este sistema mínimo seria composto por um microcomputador do tipo PC-XT, um mouse e uma tela de cristal líquido para retro-projeção.

O sistema deve poder responder perguntas e realizar operações sobre redes utilizando uma interface de fácil compreensão para o usuário professor. Esta interface deve utilizar o mínimo de operações para obter cada resultado desejado e deve utilizar o mínimo possível o teclado do computador.

As perguntas que o sistema deve ser capaz de responder automaticamente são:

Quais as marcas definidas para um lugar?

Quais as alterações definidas por uma conexão?

Quais alterações estão habilitadas?

Quais alterações estão em conflito?

Qual o caminho entre a marcação inicial e a atual?

O sistema deve ainda permitir que o professor altere o estado da rede, seja executando alterações ou alterando diretamente as marcações dos lugares.

A apresentação da rede deve ser suficientemente legível para o aluno, e para isso o sistema deve conter comandos que mostrem áreas de interesse com mais detalhe. O sistema deve escolher automaticamente qual a melhor forma de mostrar os resultados. O sistema deve fornecer ao aluno uma indicação de que o professor selecionou um comando e de que a rede está mudando de estado.

O sistema deve ser rápido o suficiente para que a demonstração não tome mais tempo da aula do que o desejado. Esta exigência refere-se ao tempo de instalação

física do hardware na sala de aula, da carga do sistema básico e das redes escolhidas, do tempo de resposta do sistema aos comandos do professor e do tempo de desmontagem física do sistema no final da aula.

As redes mostradas pelo sistema devem seguir a mesma apresentação dada no livro texto utilizado, a fim de facilitar a compreensão dos alunos e o estudo posterior. A interface gráfica deve conter todos os objetos necessários para a exibição destas redes, tais como elipses, retângulos, "fatos", arcos com setas e textos.

O software pode ficar à disposição dos alunos para estudo extra-classe.

2.2.3 Características necessárias do ambiente

O hardware utilizado deve ser o seguinte: microcomputador compatível com IBM-PC/XT (Toshiba T1000 ou equivalente), com 1 dispositivo de disco flexível; mouse compatível com Microsoft Mouse (Genius Mouse, Logitech Mouse ou equivalente); equipamento de projeção de vídeo para conexão ao computador (Data-Show ou equivalente). O tipo de vídeo utilizado normalmente é o padrão CGA (preto-e-branco, 640 por 200 pontos), mas também podem ser usados os padrões EGA (preto-e-branco, 640 por 350 pontos) e VGA (preto-e-branco, 640 por 480 pontos). O software permite a utilização sem o mouse, e a utilização de vídeo padrão "Hercules monochrome", mas esta configuração é desaconselhável.

A sala deve dispor ainda de uma tela de projeção, e possuir cortinas ou outra forma de diminuir a luminosidade, a fim de permitir a leitura das informações na tela. O contraste na tela deve ser o suficiente para permitir a leitura da tela ao mesmo tempo que haja ainda luminosidade suficiente para a leitura do quadro-negro ou outro dispositivo auxiliar.

2.2.4 Ferramentas adicionais

O software poderá permitir a modelagem na própria sala de aula. Desta forma o professor e aluno interagem mais com o sistema, explorando a sintaxe das redes e verificando os efeitos das modificações imediatamente.

2.2.5 Tipos de redes a serem modeladas

O objetivo principal do sistema é permitir a simulação de redes compactas. O sistema também poderá ser capaz de simular outros tipos de redes, tais como os sistemas condição/evento e redes lugar/transição. Para isto, o software deverá suportar as características especiais das diversas redes, tais como:

- ramos restauradores

- conexões mortas

- anotações: utilizadas para definir o universo de discurso, entre outras características.

- linguagem matemática da teoria dos conjuntos

- multiconjuntos ou "bags": usados em redes lugar/transição e em casos especiais de outros modelos.

3 FERRAMENTAS EXISTENTES

3.1 Sistemas didáticos de ensino para Redes de Petri

3.1.1 Um pacote de programas para um curso introdutório de redes de Petri

Jensen apresentou um sistema, desenvolvido no Departamento de Ciências da Computação da Universidade de Aarhus, que pode ser usado para construir, editar e analisar redes de Petri /JEN 84/.

Para comparar o sistema desenvolvido com o de Jensen, devemos levar em conta as seguintes diferenças de objetivos: o sistema de Jensen destina-se à operação pelos próprios alunos com diferentes níveis de conhecimento, realiza apenas a análise matemática de redes, e não suporta a representação gráfica de redes.

As semelhanças com o sistema de Jensen são encontradas na utilização de redes de alto nível, no tamanho das redes utilizadas, e na ausência de comandos para refinamento de redes. Ambos os sistemas são limitados pela capacidade dos computadores disponíveis, o que é uma exigência para sistemas didáticos, que devem executar em computadores "reais", e não "ideais".

3.1.1.1 Utilização de redes de alto nível

As redes utilizadas são do tipo lugar-transição. É fornecido um programa ("editor de matrizes"), através do qual o usuário introduz os nomes dos lugares e transições. Os elementos não-vazios da matriz são dados na forma de suas coordenadas e conteúdo (número do lugar e número da transição). Não é feito controle sintático dos elementos da matriz, fazendo do programa um editor de tabelas de uso geral. Os erros de sintaxe são detectados pelos outros programas do sistema, que são o "construtor de árvores" (de alcançabilidade) e o "analisador de redes de alto nível". O sistema é composto ainda por um "editor de definições", que apenas complementa a matriz com anotações.

O sistema pode ser usado com redes de até 60 nodos.

A relação entre os quatro programas é mostrada na figura 5.

Figura 3.1 Programas para curso introd. de redes de Petri

3.1.2 Analisador / Simulador de Redes de Petri

O Analisador / Simulador de Redes de Petri (ARP) é um programa para o auxílio ao projeto com redes de Petri, em desenvolvimento no LCMI (Laboratório de Controle e Microinformática da Universidade Federal de Santa Catarina) desde 1987, contando com ferramentas para redes de Petri ordinárias (redes elementares), com temporização e com temporização extendida.

O ARP foi construído em Pascal, utilizando textos e janelas. O programa roda sobre o sistema operacional DOS, em computadores compatíveis com o IBM-PC. O sistema não utiliza recursos gráficos ou mouse. Assim como o sistema de Jensen, o ARP não permite a exibição gráfica de redes de alto nível. No entanto, o sistema ARP apresenta outras qualidades que complementam o sistema desenvolvido neste trabalho, especialmente na área de análise de propriedades de redes elementares, e destina-se ao mesmo ambiente operacional. Por estes motivos o sistema ARP será analisado com mais detalhe.

a) Linguagem de descrição das redes

Os arquivos de rede são textos em formato ASCII, que contêm a descrição de uma rede de Petri. Essa descrição é feita usando-se uma linguagem específica. A linguagem de descrição de redes de Petri usada no ARP possui uma sintaxe semelhante à do Pascal, permitindo a identificação de lugares e transições por nomes. A estrutura de um texto de descrição de rede é a seguinte:

```
REDE nome_da_rede ;
        declaração de constantes
        declaração dos nodos (lugares e transições)
        declaração da estrutura (arcos)
FIM.
```

A declaração de lugares faz-se da seguinte forma (o valor entre parênteses indica a marcação inicial dos lugares declarados):

```
Robot_Parado, Torno_Operando      : LUGAR ;
Robot_Ligado, Torno_em_Espera     : LUGAR (1) ;
Pecas_Esperando                   : LUGAR (Num_Vagas) ;
```

A declaração de transições é de forma similar. É permitida a descrição da temporização da rede.

Através da declaração ESTRUTURA são descritos os arcos. Sua forma geral é:

Transição: (lugares de entrada) , (lugares de saida) ;

Essa estrutura deve ser repetida para cada transição da rede. Os nomes dos lugares são separados por vírgulas. São permitidos arcos com peso não unitário.

O sistema ARP dispõe de um compilador, que retira do texto carregado a estrutura da rede de Petri. Caso o texto apresente algum erro o compilador indica seu tipo e posiciona o cursor sobre o local onde o mesmo foi detectado, já pronto para a correção.

b) Análise das propriedades da rede

- Limitação: é calculado o número máximo de fichas (limite) em cada lugar da rede, para os estados alcançáveis. Os lugares podem ser nulos, quando nunca receberam fichas; binários, sempre possuindo uma ou nenhuma ficha; limitados, quando o número de fichas é sempre igual ou inferior a um limite finito maior que 1 ou não-limitados, quando o número de fichas tende ao infinito (simbolizado por W). Caso sejam detectados lugares não-limitados, são indicadas as seqüências de disparos de transições que levam ao crescimento de fichas nesses lugares.

- Conservação: é verificado se a soma total de fichas na rede é constante para qualquer marcação alcançável, indicando se a rede é estritamente conservativa ou não.

- Vivacidade: uma transição é viva se, a partir de qualquer estado do grafo gerado, existe uma seqüência de disparos que leve a seu disparo. Uma transição é quase-viva se foi disparada ao menos uma vez durante a construção do grafo.

- Multi-sensibilização: a enumeração de classes de estados em redes com temporização pode levar a resultados incorretos caso alguma transição esteja multi-sensibilizada no grafo. Uma transição está multi-sensibilizada se, para alguma marcação, o número de fichas na entrada da transição é maior ou igual a duas vezes o peso da entrada, para todos os lugares de entrada.

- Reiniciação: a rede é reiniciável se todos os seus estados forem reiniciáveis. Um estado é reiniciável se, partindo dele, existe alguma seqüência de disparos de transições que leve de volta ao estado inicial.

- "Live-locks": um "live-lock" é um ciclo de disparos de transições do qual a rede não possui saída, repetindo sempre os mesmos estados sem possibilidade de mudança de rumo. Caso sejam detectados "live-locks" na rede, são indicados os estados que iniciam cada um dos mesmos.

- "Dead-locks": um estado em "dead-lock" está bloqueado, não possuindo nenhuma transição sensibilizada e portanto nenhum estado sucessor. Caso sejam detectados dead-locks na rede, são indicadas as seqüências de disparos de transições que levam aos mesmos.

c) Grafo de estados

O texto do grafo de estados acessíveis indica a estrutura de um grafo que representa todos os disparos de transições que levam de um estado a outro. Cada linha deste texto é da seguinte forma:

Ex : (tm [3,7]: Ey) (tn [5]: Ez)

Isto significa que, a partir do estado Ex podemos disparar a transição tm (entre 3 e 7 unidades de tempo, no caso de rede com temporização) levando a Ey, ou a transição tn em 5 unidades de tempo, levando a Ez.

O texto dos estados acessíveis indica o conteúdo (disposição das fichas na rede) de cada estado e o domínio de disparo das transições sensibilizadas nesse estado, caso a rede possua temporização. Cada linha deste texto é da seguinte forma:

Ex : {pi, pj, 3* pk, W* pm}

D : {ta, tb [3], tc [4,9]}

Isto indica que no estado Ex os lugares pi e pj possuem uma ficha, o lugar pk possui 3 fichas e o lugar pm possui infinitas fichas (detectou-se um crescimento do número de fichas nesse lugar). No domínio de disparo, ta tem disparo permitido em [0,0], tb em [3,3] e tc em [4,9] unidades de tempo.

d) Análise de Invariantes

A análise de invariantes procura, através de cálculos sobre a matriz de incidência da rede, conjuntos de lugares ou transições com características especiais, como a conservação de fichas ou o funcionamento cíclico. Referências adicionais sobre análise de invariantes podem ser encontradas em /MAZ 90/.

Os invariantes lineares de lugar indicam as componentes conservativas da rede, ou seja, conjuntos de lugares da rede nos quais a soma ponderada do número de fichas seja constante para qualquer marcação alcançável.

Os invariantes lineares de transição indicam as componentes repetitivas estacionárias da rede, ou seja, conjuntos de transições que quando disparadas na ordem e freqüência adequada formam um ciclo, retornando ao estado de partida.

Como resultado, o programa fornece a base linearmente independente dos invariantes encontrados e todos os invariantes positivos mínimos (que não sejam super-posições de outros invariantes positivos), combinações lineares das linhas da base de invariantes.

Também é efetuado um teste de cobertura sobre os invariantes encontrados e um teste de sub-redes, indicando quais invariantes são máquinas de estados ou grafos de eventos, conforme sua topologia.

Um invariante linear de lugar pode ser uma máquina de estados se cada transição ligada a um lugar do invariante possui somente um arco de entrada e um arco de saída, ambos de peso unitário.

Um invariante linear de transição pode ser um grafo de eventos se cada lugar ligado a uma transição do invariante possui somente um arco de entrada e um arco de saída, ambos de peso unitário.

e) Verificação

A verificação por equivalência de linguagem permite observar em detalhe o comportamento de uma rede de Petri, enfocando a atenção em algumas partes da rede e abstraindo as demais. Ela consiste em considerar um conjunto de transições da rede como sendo eventos "visíveis", sendo as demais consideradas invisíveis, e obter um autômato finito determinístico mínimo que indica o seqüenciamento entre os eventos considerados visíveis.

O usuário define quais transições da rede são visíveis e o programa, a partir do grafo de estados acessíveis, fornece como resultado o autômato representando o sequenciamento entre as transições visíveis e também os caminhos (seqüências de disparo) possíveis nesse autômato.

f) Simulação

O módulo de simulação permite acompanhar a evolução do funcionamento de uma rede, escolhendo as transições a disparar e observando o estado da rede e sua evolução, a qualquer instante.

Na janela maior há um diagrama da trilha percorrida pela simulação até o momento. No lado esquerdo da trilha é indicado o nome de cada estado, se memorizado, ou sua profundidade em relação ao estado inicial. No lado direito é indicado o nome de cada transição disparada.

Os números junto da trilha indicam o número de transições ainda não disparadas em cada estado. O estado atual é o da extremidade inferior da trilha.

Quando há bloqueio, a trilha termina em uma barra horizontal. Quando o estado gerado por um disparo for duplicata de algum estado já percorrido, este é indicado por uma linha tracejada, e o simulador permanece no estado onde ocorreu o disparo.

Os comandos disponíveis no simulador são:

- Texto de evolução: liga ou desliga a geração do relatório de simulação, que contém um histórico detalhado da simulação efetuada.

- Memória de estados: permite memorizar estados para retorno posterior. Um estado pode ser memorizado com qualquer nome ainda não utilizado. No máximo 150 estados podem ser memorizados.

- Disparo de transições: das transições sensibilizadas no estado atual pode ser escolhida qualquer uma para disparar. As transições ainda não disparadas são as mais claras na lista. Caso o instante de disparo permitido para a transição não seja definido, será perguntado ao usuário.

- Retorno: permite fazer o retorno a um estado anterior ("back-tracking"), enquanto houverem estados anteriores na trilha de evolução. Se a trilha estiver vazia e a rede não possuir temporização, a lista das transições disparáveis "em reverso" é apresentada, podendo ser disparada "em reverso" qualquer transição.

- Lugares em observação: permite escolher quais lugares da rede terão sua marcação indicada na tela a cada modificação de marcação.

- Edição do estado atual: permite alterar a marcação ou o intervalo dinâmico de disparo de cada transição sensibilizada no estado atual. Alterando-se algum valor a trilha de estados anteriores é apagada.

- Visualizar estado: permite visualizar um estado da trilha, bastando para isso informar a profundidade do estado desejado.

g) Avaliação de Desempenho

Este módulo trabalha com redes de Petri com temporização extendida , que possuem uma função de densidade de probabilidade associada ao intervalo de disparo de cada transição.

Cada ciclo de simulação consiste no disparo das transições sensibilizadas em instantes aleatórios (sorteados de acordo com o intervalo de disparo e a curva de densidade de probabilidade de cada transição) até alcançar um destino, um bloqueio ou superar o número máximo de disparos fixado pelo usuário.

A avaliação produz um texto contendo as medidas estatísticas efetuadas sobre a rede:

- Tempo médio (e desvio padrão) de acesso a cada destino.

- Tempos mínimo e máximo de acesso detectados a cada destino.

- Probabilidades de acesso relativas entre destinos; número de acessos efetuados a cada destino; número médio de disparos de cada transição por ciclo de simulação.

- Marcação média registrada nos lugares, por unidade de tempo.

- Tempo médio de espera de cada transição para disparar, a partir de seu instante de sensibilização.

- Número de ciclos improdutivos, onde não foi alcançado nenhum destino.

h) Impressão de Resultados

3.1.3 Outros sistemas de propósito não didático

PROTEAN /BIL 88/ é um sistema desenvolvido para a análise de protocolos. A ferramenta inclui a edição e análise de redes de Petri numéricas, em modo gráfico (a saída de resultados também é gráfica), simulação interativa, análise de alcançabilidade, análise de grafos direcionados e possui um sistema de ajuda on-line. O sistema necessita, no entanto, de um computador com sistema operacional VMS ou UNIX.

DESIGN/CPN /ALB 89/ é um sistema desenvolvido pela empresa Meta Software Corporation, que apresenta características interessantes para a utilização de redes de Petri coloridas (CPN). As redes deste tipo assemelham-se às que são objetivo deste trabalho. O sistema DESIGN/CPN não permite a existência de ramos restauradores ou de conexões mortas (descritos em /HEU 90/) e seu arquivo de descrição de redes não possui documentação aberta. Hintz /HIN 90/ estudou a possibilidade de produção de ferramentas utilizando um outro produto, o DESIGN/OA, que apresenta arquitetura de programação mais aberta. O sistema DESIGN/OA exige, além da programação dos procedimentos relacionados à rede, que o computador disponha de um ambiente gráfico do tipo X-Windows ou Microsoft Windows.

A implementação das redes compactas /HEU 90/ no DESIGN/OA exigiria a definição de toda a linguagem de anotação específica destas redes, e ainda assim não permitiria a definição de conexões mortas. O software obtido seria ineficiente, pois teria que executar a simulação simbólica da rede. Além disso, o sistema possui uma interface que, embora seja bastante adequada a utilização por um único usuário, não apresenta as qualidades necessárias para a apresentação em sala de aula, tais como a boa legibilidade dos caracteres.

Embora o sistema DESIGN/OA não pareça adequado à utilização em sala de aula, ele pode vir a ser útil na construção de uma ferramenta de edição de redes. Estas redes, se for possível obter uma descrição de sua estrutura, poderão ser utilizadas com o sistema de que trata este trabalho.

3.2 Restrições feitas sobre as redes tradicionais

O sistema teórico básico é formado pelas redes condição/evento (C/E). Este sistema foi proposto por Petri como um modelo de referência da teoria de redes /PET 76/.

A definição tradicional de redes de Petri não permite a existência de lugares ou transições isolados. O sistema implementado não tem esta limitação, mas o usuário modelador deve considerá-la como existindo, se quiser manter a compatibilidade com as Redes tradicionais.

As redes C/E são marcadas utilizando-se "marcas elementares", que são representadas graficamente como pontos pretos. O usuário modelador pode representar estas marcas através do caractere ASCII "~", que aparece graficamente como um ponto preto para o usuário final. O número máximo de marcas (elementares ou não) em um lugar é limitado na implementação (devido à otimização para o hardware utilizado).

As redes C/E utilizam também anotações que identificam as condições. Estas anotações podem também ser feitas utilizando a linguagem de modelagem do sistema implementado. O sistema também suporta a definição de constantes, variáveis, predicados e fórmulas de primeira ordem.

3.2.1 Redes predicado-transição

As redes predicado/transição (PrT) foram propostas por Genrich e Lautenbach /GEN 81/. As redes PrT utilizam grupos de condições (das redes C/E) para construir lugares. Os eventos são identificados anotando os arcos com constantes. As constantes são substituídas por variáveis dos lugares para obter a condição correspondente. Alguns dos tokens também são representados por constantes.

A linguagem de anotação descrita em /GEN 87/ permite a utilização de símbolos matemáticos ($\supset$, $\exists$, ϕ) que não podem ser representados pelo sistema implementado. No entanto, o usuário pode utilizar nomes em formato ASCII para os símbolos.

As redes PrT têm uma legenda, onde localiza-se a definição do universo de discurso. O sistema implementado suporta legendas. No entanto, a experiência mostrou que as legendas dificultam a leitura da rede devido à baixa resolução gráfica do hardware utilizado. A eliminação da legenda, que pode ser feita se o usuário modelador considerar necessário, não prejudicará o funcionamento da rede. Caso o modelador preferir utilizar legendas, mas ocupando apenas uma pequena área da tela gráfica, o usuário final receberá o início da mensagem, seguida do símbolo "...", indicando que pode selecionar aquela área para visualização (zoom).

O sistema implementado não permite a representação de "todos elementos de um conjunto" (presentes em um lugar) através do nome do conjunto. Por isso, mesmo que todos os elementos estivessem presentes, eles seriam representados individualmente (separados por vírgula). A anotação de um arco não tem esta limitação, que só vale para o conteúdo dos lugares.

A representação de fatos (transições mortas, conexões mortas) é possível e existe um símbolo específico na linguagem de descrição utilizada pelo sistema implementado.

3.2.2 Redes coloridas

Em /JEN 87/, Jensen utiliza as diferenças existentes entre as redes PrT e as Redes Coloridas (CP) para definir informalmente as redes CP. As diferenças apontadas são as seguintes:

1) O conjunto de cores de tokens possíveis em um lugar é explicitamente definido. Esta exigência é também uma exigência do sistema implementado.

2) O número de tokens adicionados ou removidos de um dado lugar pode ser diferente para duas cores durante a mesma transição. Isto acontece também no sistema implementado.

3) O conjunto de expressões e predicados permitidos não é explicitamente definido (mas é desejável que pertençam a uma álgebra, de onde todas as expressões, predicados, funções e conjuntos possam ser construídos). Da mesma forma, o sistema implementado permite o uso de álgebras não explicitamente definidas. A limitação imposta a esta regra é que o alfabeto utilizado pela notação desta álgebra seja o alfabeto ASCII. Com exceção desta limitação, e mais a limitação de hardware (memória e tempo de processamento), o sistema permite a definição de uma álgebra pelo usuário modelador.

O sistema implementado representa os seguintes conceitos sobre redes CP e PrT: conflitos, concorrência, estados iniciais e fatos. Além disso, o sistema pode procurar um caminho (definido a partir da teoria dos grafos) entre duas marcações, e identificar pré- e pós-conjuntos para uma dada marcação. O sistema pode determinar se uma marcação é alcançável, mas não pode, na maioria dos casos, provar que uma marcação não seja alcançável.

Quanto à sua forma, a rede modelada pode estar ou não na forma normal.

O sistema aceita a definição de um conjunto de variáveis com tipos, expressões de arco com variáveis livres e "guardas" (expressões associadas a transições). O sistema modela a forma gráfica das Redes, não podendo ser representada a forma matricial. Segundo Jensen /JEN 87/, a forma gráfica é normalmente utilizada para descrição de sistemas e para explanação informal, enquanto a forma matricial é aplicada na análise formal.

4 INTERFACE COM O USUÁRIO

4.1 Justificativa da escolha da interface com o usuário

A interface com o usuário visa permitir que o professor utilize o mínimo possível o teclado do computador. O ambiente de trabalho é normalmente escuro, para permitir a visualização da imagem do equipamento DataShow, que apresenta pouco contraste. Além disso, o professor precisaria olhar para o teclado, desviando a atenção do aluno. Por isso todos os comandos foram implementados através da utilização do mouse.

Todos comandos sobre a rede são executados em uma tela que apresenta uma região da rede em detalhe, a descrição matemática do estado daquela entidade que está sendo mostrada e, em uma janela, uma visão geral da rede. A interface apresenta também a opção de visualização total da rede. Quando o professor aponta para uma região da rede, o sistema mostra aquela região com mais detalhe. O nível de detalhamento da apresentação (área de interesse) faz parte do arquivo de descrição da rede, mas o sistema permite que o professor solicite detalhamento adicional.

A comunicação com o usuário é feita através da apresentação do desenho da rede e da apresentação de menus. O usuário age sobre o sistema apontando uma área na tela. Somente quando o usuário faz a seleção de uma nova rede deve digitar o nome da rede desejada através do teclado. Para todos os demais comandos, apenas o mouse é utilizado. O sistema permite, com severas limitações, o funcionamento sem o mouse. O menu é apresentado somente quando ele é indispensável, para não prejudicar a visualização da rede.

Devido a limitações do software básico utilizado, o sistema redesenha a tela cada vez que uma alteração deve ser feita na apresentação. O usuário não tem um retorno sobre a função escolhida com o mouse, que permita verificar se a função escolhida é efetivamente a desejada.

4.2 Interação do usuário com o sistema

4.2.1 Entrada no sistema

Para executar o programa a partir do sistema operacional MS-DOS, o usuário deve digitar o comando

RPaa bbbbb /H

onde RPaa é o nome do programa, aa é o número da versão, bbbbb é o nome de um arquivo de descrição de rede (opcional) e /H indica o modo gráfico de alta resolução (opcional).

A opção de modo gráfico de alta resolução significa utilizar a maior resolução disponível (640 x 480 VGA ou 640 x 350 EGA). Se não utilizada, o sistema entra em modo 640 x 200 pontos, compatível com Datashow CGA ou com o computador Toshiba T1000. Em um microcomputador com vídeo "Hercules", existe somente a resolução 720x350.

O programa reconhece automaticamente o tipo de controlador de vídeo que está instalado no computador. O modo /H só tem efeito para controladores do tipo EGA e VGA.

Exemplo: RP20 RP4-6.DAT /H

- Executa a versão 20 do simulador, carregando a rede de nome RP4-6, utilizando o modo gráfico de resolução mais alta disponível.

O programa apresentará a rede em modo "tela inteira", como mostra a figura 6.

4.2.1.1 Correção de problemas

a) A tela aparece vazia

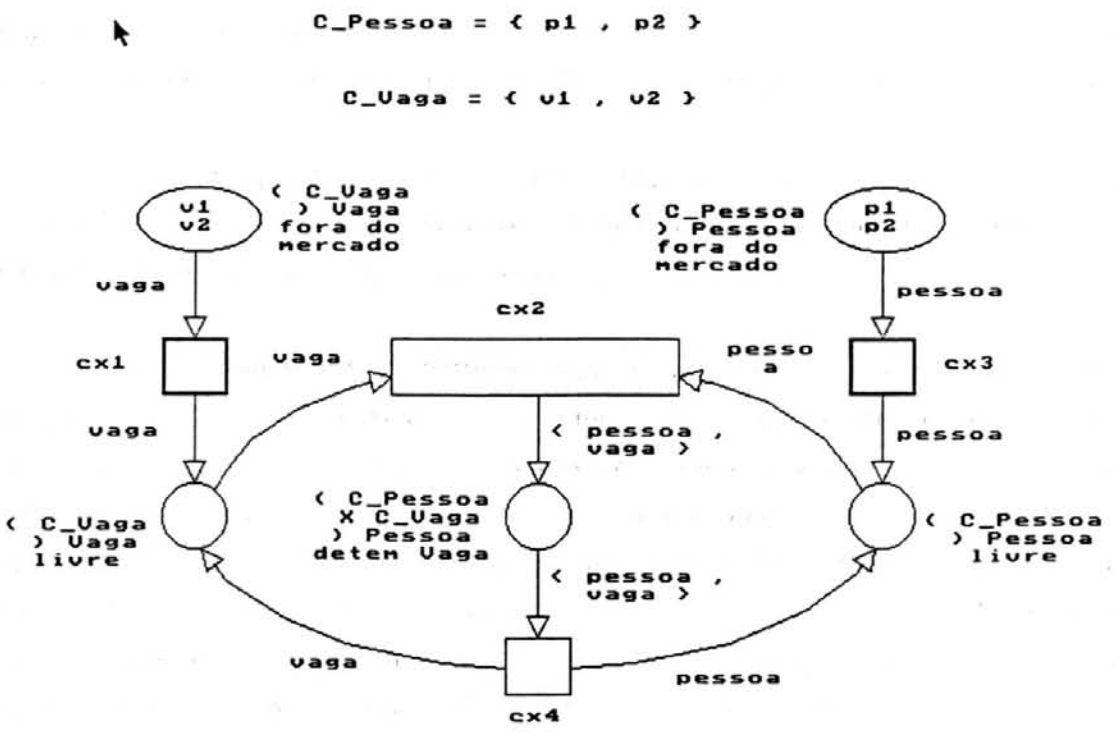


Figura 4.1 Tela inteira

Em caso de erro na linha de comando, a tela aparecerá vazia. Apenas a borda da tela será desenhada. Neste caso, o usuário poderá pressionar o botão 2 do mouse, solicitando o menu de comandos. Ao aparecer o menu, o usuário pode mover a seta para opção desejada e pressionar o botão 1 do mouse.

Duas opções serão úteis: "Lê nova rede" permite selecionar uma rede que esteja no disco; "Fim" permite voltar ao sistema operacional.

A causa mais freqüente de erros nesta etapa inicial é o esquecimento de colocar a terminação ".DAT" ao digitar o nome de uma rede. O *nome* de uma rede é aquele que aparece no diretório do disco ao utilizar-se o comando "DIR" do sistema DOS.

b) A seta do mouse não se move

Existem diversos problemas que podem causar a parada do mouse. O usuário deve verificar a conexão física (cabos, sujeira no mecanismo do mouse), e software (presença do "driver" do mouse e sua instalação). O software já foi testado com mouses das marcas Microsoft, Logitech, Genius e Clix. Foi detectado um problema de incompatibilidade entre drivers de mouse e a placa gráfica Hercules. Este problema foi resolvido dentro do programa, mas é possível que ainda aconteçam incompatibilidades em outras combinações de mouses e placas gráficas. O programa pode ser usado sem a presença do mouse, como é descrito na seção seguinte.

c) "Erro: conversão para inteiro" ou "Erro: símbolo não definido (...)"

Estes erros indicam que o arquivo de descrição da rede foi danificado. Normalmente é muito difícil recuperar este arquivo manualmente. Por isso, conserve sempre uma cópia de segurança (backup) dos arquivos de redes.

4.2.2 Utilização do programa sem o mouse

A utilização do programa sem o mouse não é recomendada, mas é possível.

Caso não apareça a seta do mouse na tela, pressione a tecla PgDn. O programa fornecerá um cursor quadrado, que substitui a função da seta indicadora do mouse.

O movimento da seta indicadora é realizado com as setas do teclado.

Os botões do mouse são substituídos pelas teclas:

- Home Botão 1 (seleciona uma função)

- PgUp Botão 2 (chama o menu do sistema)

4.2.3 Menu do sistema

A figura 7 apresenta o menu do sistema, que é utilizado para as operações com o simulador e para a comunicação com arquivos.

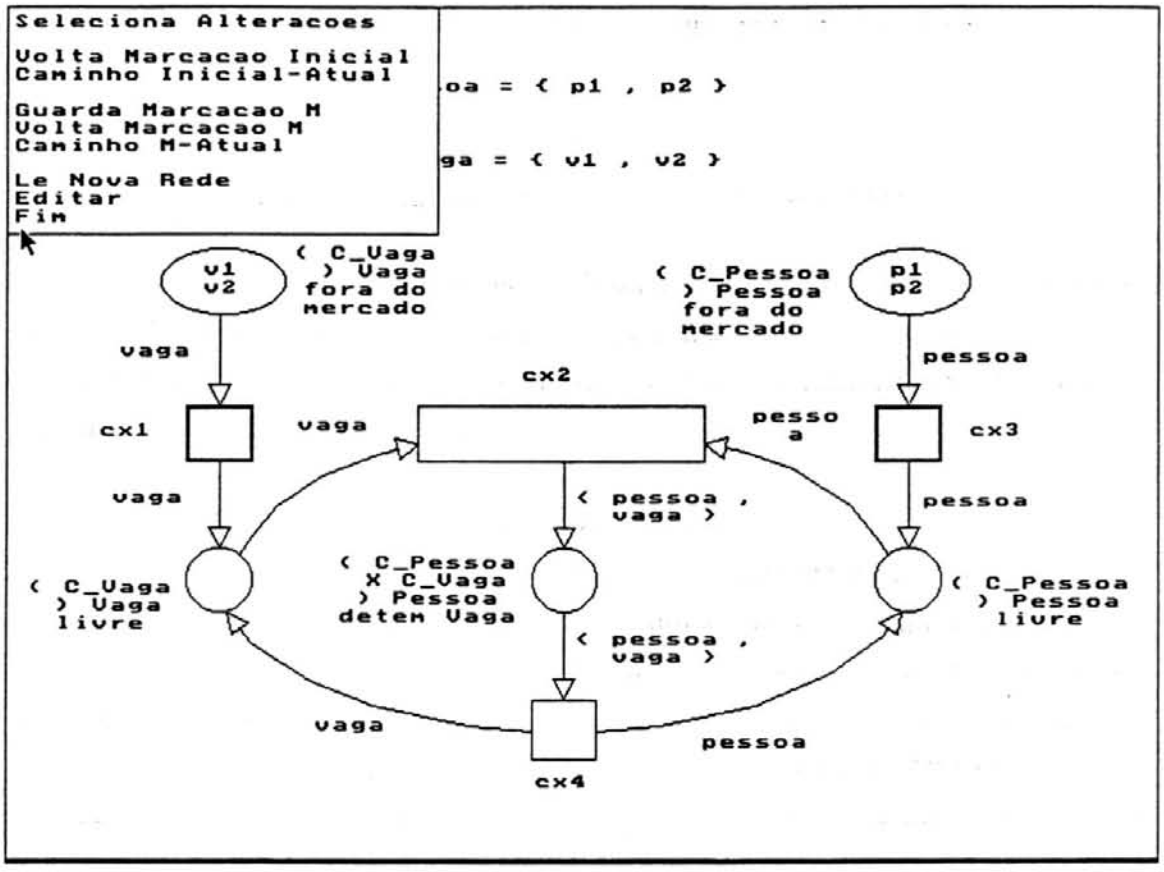


Figura 4.2 Tela com menu

O menu aparece somente quando o botão 2 do mouse é pressionado.

Para sair do menu sem executar nenhum comando, o usuário deve pressionar o botão 1 do mouse com a seta fora da área do menu.

Para executar um comando, o usuário pressiona o botão 1 do mouse com a seta sobre o comando desejado. Os comandos são:

a) Seleciona Alterações

Disponível quando o simulador está em modo de "execução imediata", ou seja, cada alteração indicada é imediatamente realizada.

Ao executar, este comando passa o simulador para modo "armazena passo", ou seja, cada alteração indicada é armazenada. As alterações armazenadas são executadas depois, de forma concorrente.

b) Executa Passo

A opção "Executa Passo" (figura 8) está disponível quando o simulador está em modo de "armazena passo", faz com que as alterações armazenadas sejam executadas. Em seguida, o programa volta para o modo "execução imediata".

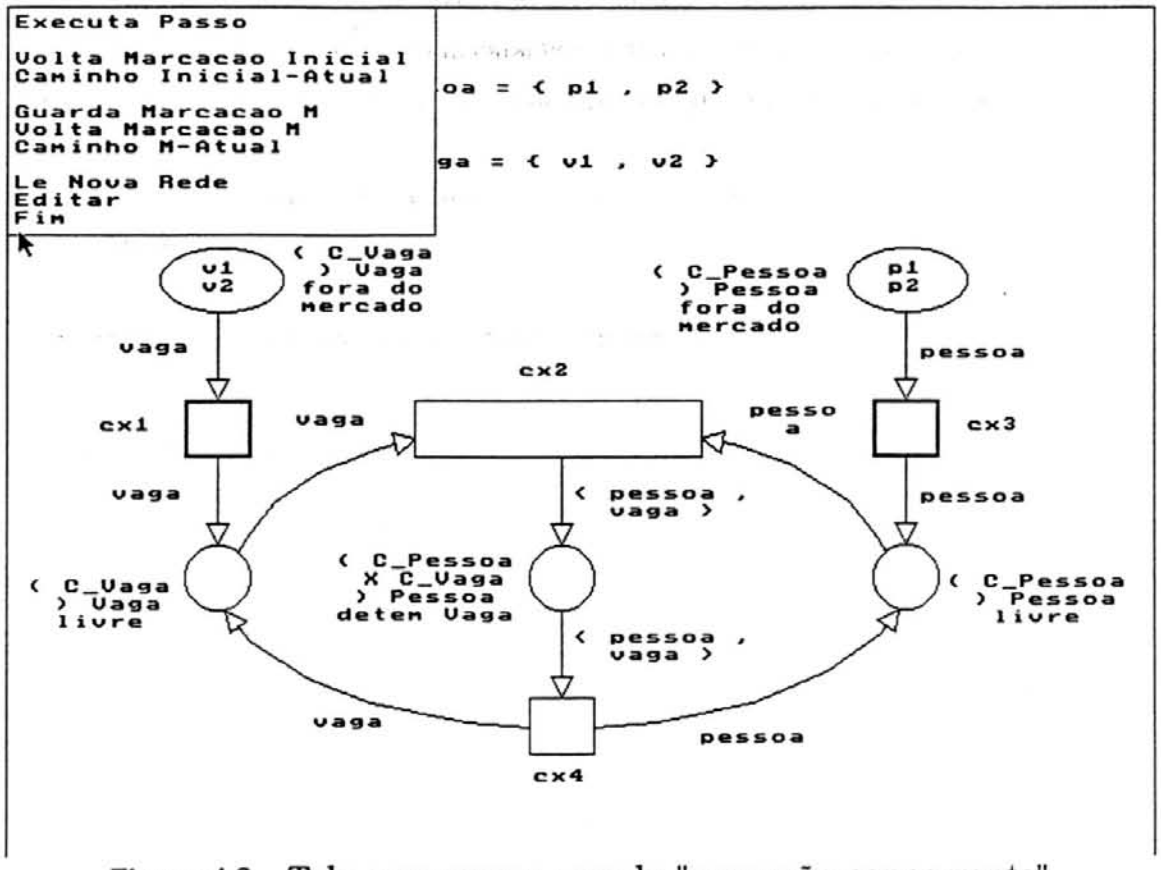


Figura 4.3 Tela com menu - modo "execução concorrente"

É importante notar que esta opção não estará disponível caso a opção "Seleciona Alterações" esteja habilitada.

c) Volta Marcação Inicial

Volta ao estado em que a rede estava quando foi lida do disco.

d) Caminho Inicial-Atual

Disponível se a rede não tem conexões mortas.

Obtém e mostra o caminho mais curto entre a marcação inicial e a atual.

e) Guarda Marcação M

Memoriza o estado da rede, permitindo que mais tarde o usuário possa voltar a este estado. Também permite que mais tarde o usuário solicite a obtenção do caminho entre este estado e um outro.

f) Volta Marcação M

Volta a rede ao estado memorizado anteriormente.

g) Caminho M-Atual

Disponível se a rede não tem conexões mortas.

Obtém e mostra o caminho mais curto entre a marcação memorizada e a marcação atual.

h) Lê Nova Rede

Lê uma rede do disco.

É apresentado um quadro com a pergunta "Nome:". O preenchimento do nome é obrigatório. Se a rede não existe no disco, o estado do programa não é alterado e a rede atual continuará na memória.

i) Editar

Entra em modo de edição para a rede atual (figura 9).

O processo de edição e compilação de redes será descrito no capítulo 7.

j) Fim

Encerra o programa e volta para o sistema DOS.

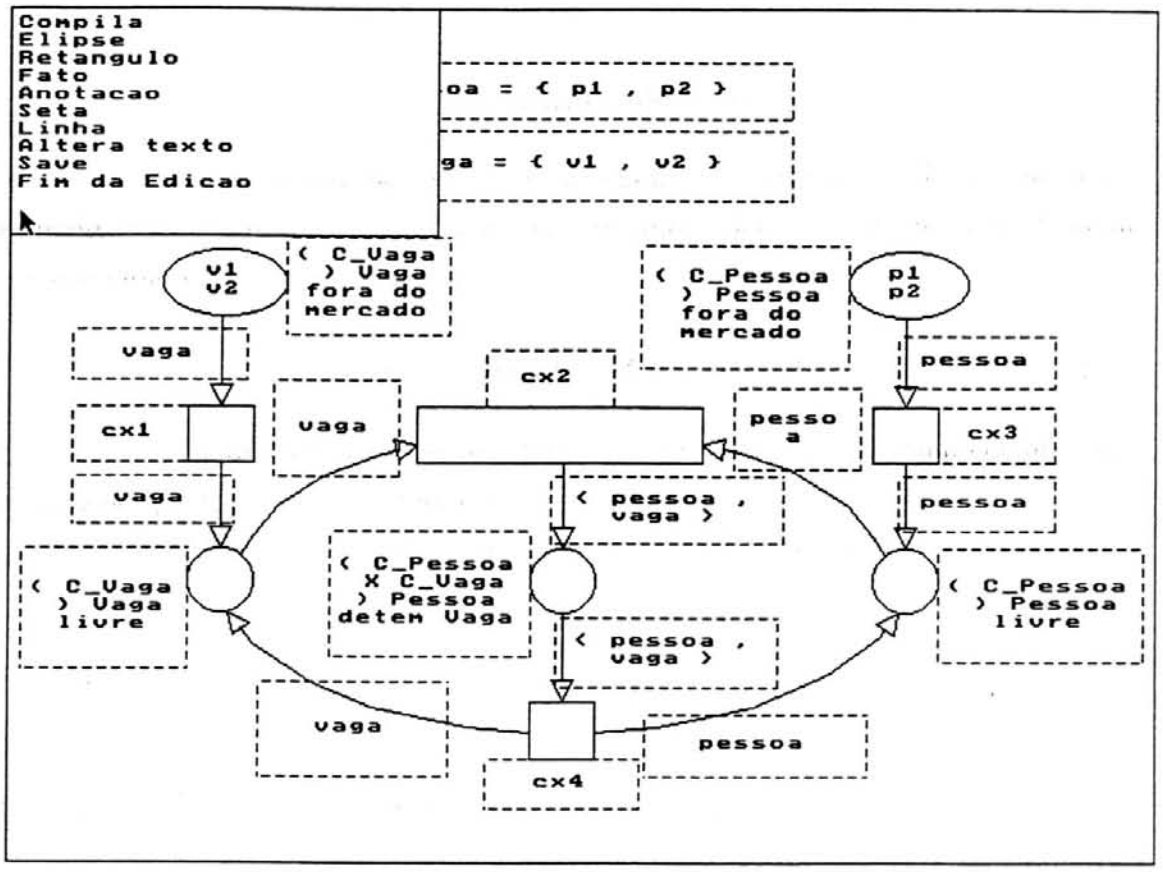


Figura 4.4 Menu do editor

4.2.4 Operação fora do menu

Fora do menu, o usuário tem duas opções básicas de visualização: em tela inteira (figura 6) e em região de interesse (figura 10).

a) Visualização em tela inteira

Em tela inteira, o sistema mostra a área total da rede, mostrando com linhas mais grossas as alterações habilitadas. Todas anotações da rede são mostradas. Quando o espaço reservado a uma anotação não permite que ela seja mostrada completa, o sistema mostra o sinal "..." no final do texto correspondente. As anotações são centralizadas nas áreas reservadas. O tamanho da rede em relação à tela é calculado pelo sistema, de acordo com o hardware disponível e com a opção selecionada pelo usuário ao entrar no sistema.

Funções definidas:

- Apontando para uma região da rede e pressionando o botão 1 do mouse, o sistema entra em modo de região de interesse.

- Pressionando o botão 2 do mouse é apresentado o menu do sistema.

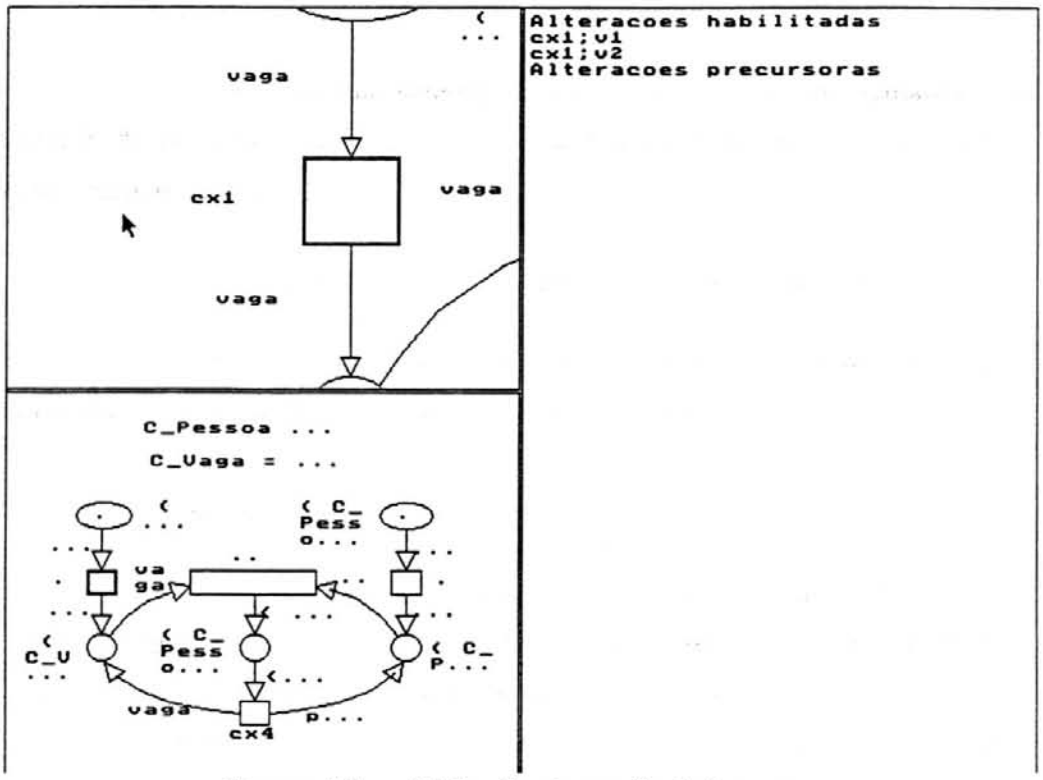


Figura 4.5 Tela da área de interesse

b) Visualização de região de interesse

Neste modo (figura 10) aparecem três janelas simultâneas, contendo informações sobre:

- onde situa-se a região de interesse, dentro da rede ("mapa" resumido da rede, em modo gráfico);

- conteúdo da região de interesse ("zoom", em modo gráfico);

- descrição do estado lógico do lugar ou conexão situado no centro da região (em modo texto).

Cada uma destas janelas tem *funções* que podem ser ativadas pelo usuário:

- Na janela de situação geral ("mapa"), o usuário visualiza toda a rede e, em linhas mais grossas, a região selecionada. Ao pressionar o botão 1 do mouse com a seta sobre uma outra região da rede, aquela região torna-se a região de interesse. Caso seja selecionada uma área vazia desta janela, volta-se à visualização da rede em tela inteira.

- Na janela de região gráfica ("zoom"), o usuário pode selecionar uma entidade e obter uma ampliação ainda maior da área apontada. A seleção de entidades em qualquer uma das janelas gráficas não altera o estado da rede.

- A janela de texto é utilizada para alterar o estado da rede. O usuário utiliza o relatório presente na janela de texto da seguinte forma: Para uma área de lugar (elipse), o texto descreve a marcação atual e diz quais marcas, entre as definidas para o lugar, não estão presentes. Selecionando o texto descritivo de uma marca, o usuário pode inverter seu estado, tornando-a em vigor ou não. Para uma área de alteração (retângulo), o texto descreve as alterações habilitadas e diz quais as alterações predecessoras. Selecionando uma alteração habilitada, a alteração acontece na rede. Selecionando uma alteração predecessora, a rede "funciona para trás". Junto às alterações habilitadas, o sistema mostra outras alterações com que esta alteração tenha conflito. Através do menu (botão 2 do mouse) o usuário pode mudar o estado do sistema para "Selecionar Alterações". Neste modo, cada alteração selecionada entra para uma lista de alterações. A lista é executada de forma "concorrente" ao ser selecionado o comando "Executa Passo" do mesmo menu.

4.3 Observações

A maior parte das funções disponíveis no programa está sempre ao acesso do usuário ao simples toque de um botão do mouse. O usuário deve apenas observar a tela apresentada, já que não existe sempre no sistema um menu tradicional, com todos os comandos.

De forma geral, as telas gráficas controlam a *apresentação gráfica* da rede, enquanto as telas de texto controlam as *modificações da estrutura interna* da rede ou *modificações do estado* da rede.

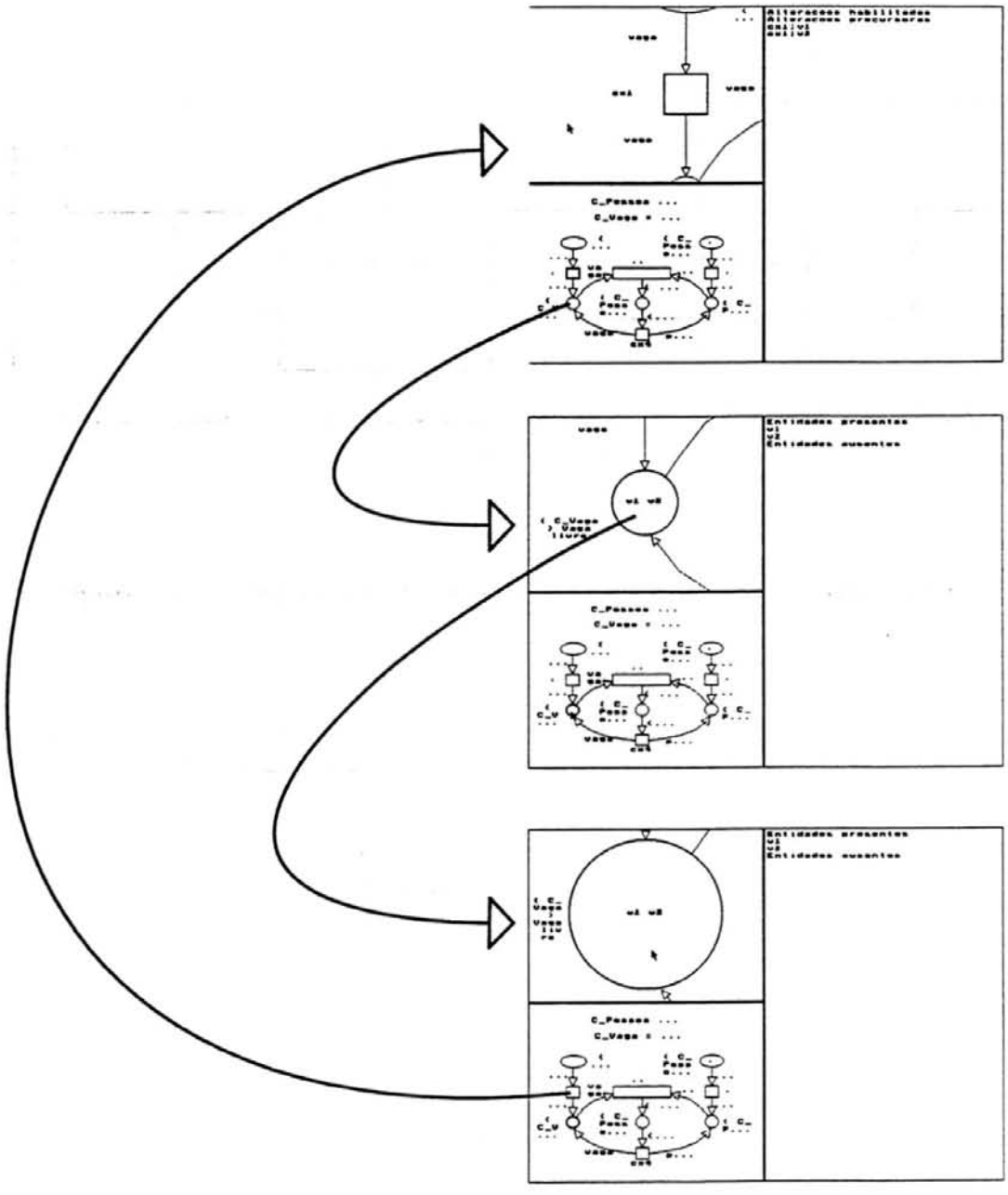


Figura 4.6 Funcionamento da tela de área de interesse

5 LINGUAGEM DE DESCRIÇÃO DE REDES DE ALTO NÍVEL

5.1 Justificativa da escolha da linguagem de descrição

O sistema recebe a descrição de uma rede em formato texto. A forma textual foi escolhida por ser de fácil implementação, por permitir a localização de erros mais facilmente, por ser legível, e por tornar desnecessária a implementação de um editor específico.

Outras alternativas para fazer a descrição seriam através de um editor gráfico ou através de uma outra linguagem textual de mais alto nível. Estas alternativas são viáveis através da construção de tradutores ("filtros") que convertam arquivos gráficos ou de outra linguagem para a linguagem utilizada por este sistema. Não foram implementados até o momento tradutores deste tipo, mas foi estudado o desenvolvimento de um editor baseado em estação de trabalho SUN, utilizando o software DESIGN/OA /HIN 90/. Este desenvolvimento foi abandonado após a constatação de que o sistema DESIGN/OA não apresentava documentação suficiente para que se fizesse a conversão de arquivos. O capítulo 7 apresenta o estado atual de desenvolvimento das ferramentas de descrição de redes.

A linguagem de descrição é baseada na descrição formal de Redes de Petri e permite a descrição das propriedades de uma rede do ponto de vista matemático, através de conjuntos, e do ponto de vista do usuário, através de símbolos gráficos. Redes de Petri de alto nível são representadas através da composição de símbolos básicos.

Pode ser necessário um grande número de linhas de programa para representar uma rede de alto nível, e uma quantidade proporcional de memória principal do computador, limitando o tamanho das redes que podem ser armazenadas.

A linguagem de descrição funciona como uma linguagem "assembly" para o sistema, sendo utilizada também para controle interno das funções de simulação, leitura de arquivos, execução de comandos do mouse e comandos do menu. O sistema possui um núcleo que interpreta esta linguagem, e que recebe comandos tanto do arquivo de descrição da rede como do módulo de interface com o usuário.

As redes utilizadas como exemplos foram desenhadas inicialmente utilizando o software AutoCAD, e traduzidas manualmente para a linguagem de descrição. A parte matemática da descrição foi feita manualmente. A partir da versão 20 do software, as redes passaram a ser descritas utilizando o editor/compilador interno do programa.

5.2 Descrição da linguagem

A linguagem de descrição utilizada para a entrada de redes no sistema é composta por sentenças e comentários. Os comentários são indicados por um ponto-e-vírgula. As sentenças iniciam com um caractere indicativo do comando, e continuam com as opções relativas aquele comando. As sentenças terminam com o fim da linha física.

Os comandos definem os nomes das entidades, a dinâmica da rede e a geometria da rede.

5.3 Nomes de entidades e dinâmica da rede

a) Nomes de entidades

M*<número da marca>* *<nome da marca>*

Quando se deseja uma marca elementar, *<nome da marca>* deve ser o caractere til (~).

exemplo: M1 a1
M2 a2

marca elementar: M3 ~

b) Marcas definidas para um lugar

L*<número do lugar>* *<lista de números de marcas>*

exemplo: L1 1 2

c) Alterações definidas

A*<número da alteração>* *<número da conexão>* *<descrição textual>* *<lista de alterações>*

A lista de alterações é composta de conjuntos ordenados com três elementos contendo: *<número do lugar, marcação anterior, marcação alcançada>*.

Quando se deseja definir uma porta de saída, *<marcação anterior>* deve conter o número da marca, e *<marcação alcançada>* deve ser zero. Para definir uma porta restauradora de saída, *<marcação anterior>* deve conter o número da marca, e *<marcação alcançada>* deve ser menos um. Quando se deseja definir uma porta de entrada, *<marcação anterior>* deve conter zero, e *<marcação alcançada>* deve conter o número da marca. Para definir uma porta restauradora de entrada, *<marcação anterior>* deve conter menos um, e *<marcação alcançada>* deve conter o número da marca.

exemplo: A1 5 2 0 1 3 1 0

No exemplo é definida uma alteração número um, sobre a conexão 5 (R5). Quando o lugar 2 não contém a marca 1 e o lugar 3 contém a marca 1, a alteração está habilitada. Quando a alteração ocorre, a marca 1 sai do lugar 3 e aparece no lugar 2.

exemplo: A2 3 4 5 -1 7 8 0

No exemplo é definida uma alteração número dois, sobre a conexão 3 (R3). Quando o lugar 4 contém a marca 5 e o lugar 7 contém a marca 8, a alteração está habilitada. Quando a alteração ocorre, a marca 8 sai do lugar 7. O lugar 4 permanece inalterado.

d) Marcação inicial

I*<número do lugar>* *<lista de números de marcas>*

exemplo: I1 1 2

5.4 Geometria da rede

Todas as coordenadas utilizadas são adimensionais, inteiras, na faixa de -32000 a +32000. O sistema muda automaticamente a escala do desenho de acordo com o hardware disponível e com a área selecionada.

a) Lugar

E*<número do lugar>* *<coordenadas>* *<área de interesse>*

As coordenadas são as do retângulo circunscrito na elipse. São dadas primeiro as coordenadas do ponto superior esquerdo, e depois as coordenadas do ponto inferior direito.

exemplo: E1 10 10 100 20 5 5 105 25

define uma elipse representando o lugar 1, na posição compreendida entre as posições (x=10, y=10) e (x=100, y=20). Ao selecionar a região de interesse deste lugar, o professor obterá uma visualização da área compreendida entre os pontos (x=5, y=5) e (x=105, y=25).

b) Conexão, fato, anotação

Conexão:

R*<número da conexão>* *<coordenadas>* <área de interesse> <anotação>

Fato:

F*<número da conexão>* <coordenadas> <área de interesse> <anotação>

Anotação:

N*<número da anotação>* *<coordenadas>* <área de interesse> <anotação>

São dadas primeiro as coordenadas do ponto superior esquerdo, e depois as coordenadas do ponto inferior direito.

exemplo: F1 10 10 100 20 5 5 105 25 v1 = v2

define um "fato", na posição compreendida entre as posições (x=10, y=10) e (x=100, y=20), contendo a anotação "v1 = v2". Ao selecionar a região de interesse deste lugar, o professor obterá uma visualização da área compreendida entre os pontos (x=5, y=5) e (x=105, y=25).

c) Linha, seta

Seta:

X<*número da seta*> <coordenadas>

Linha:

Z<*número da linha*> <coordenadas>

As pontas das setas serão desenhadas sempre do mesmo tamanho, independentemente do nível de detalhamento da visualização. A ponta da seta localiza-se na segunda coordenada dada.

exemplo: X1 10 10 20 20

define uma seta com ponta em (x=20, y=20). Linhas complexas são definidas através da conexão de várias linhas simples, independentes.

5.5 Usabilidade da linguagem

Quanto à linguagem do sistema, o usuário deve talvez levar em conta a expressão utilizada por Jensen /JEN 87/ (sobre a transformação de Redes CP em Matrizes CP): "The definition of the two translations may seem a bit complicated - when they are given in their general form presented above. However, after a few times of exercise the translations can be done fast and nearly 'without thinking'.".

No capítulo 7 são apresentadas algumas ferramentas que vêm sendo implementadas com a finalidade de facilitar o trabalho de descrição de redes para este software. A versão 20 do programa já apresenta um editor simples, e um compilador que aceita a sintaxe completa das redes compactas. As duas ferramentas foram implementadas dentro do próprio software, e estão disponíveis a qualquer momento durante a simulação, permitindo a criação ou alteração de redes compactas.

5.6 Um exemplo mais detalhado

A figura 12 apresenta um exemplo de rede de Petri compacta com notação especial para tratamento de todas entidades de um lugar, adaptado de /HEU 90/.

A rede é explicada da seguinte forma: Vamos imaginar que um autor possa submeter a uma conferência mais de um artigo. É necessário permitir a recepção do segundo artigo, que seria desabilitada pela presença do autor no lugar *autores*. Para dar

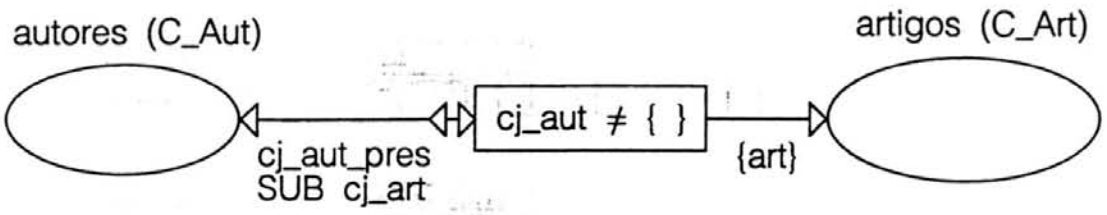


Figura 5.1 Exemplo de rede com notação especial

valores, considerem-se as alterações *rec;ar1,{au1,au2}* e *rec;ar2,{au1}* . A primeira corresponde à recepção do artigo *ar1* de autoria de *au1* e de *au2* e a segunda corresponde à recepção do artigo *ar2* de autoria de *au1*. Ambas estão em conflito pela marca de saída *(autores,au1)*.

Para contornar esta situação, podemos utilizar a notação para tratamento de todas marcas de um lugar, expressando que, quando da recepção de um artigo, devem ser incluídos apenas os autores ausentes do artigo. O ramo de tratamento de todas as entidades é de saída, expressando a inclusão de entidades.

5.6.1 Definição da geometria de rede

O desenho da rede é feito utilizando-se coordenadas arbitrárias. As figuras são numeradas, como mostram as figuras 13 e 14.

A partir das coordenadas é gerada a lista de figuras que compõem a rede:

C1 2 2 4 4 0 0 12 5

onde

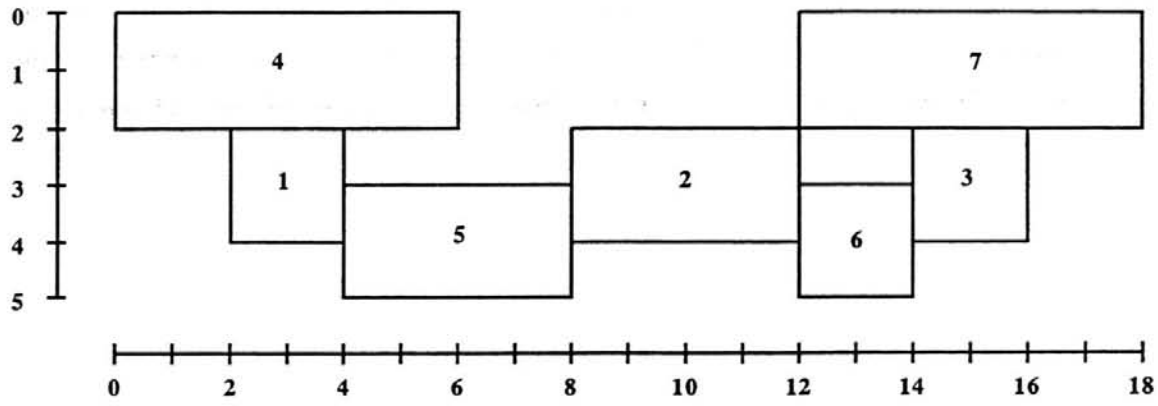


Figura 5.2 Conversão do desenho da rede para coordenadas

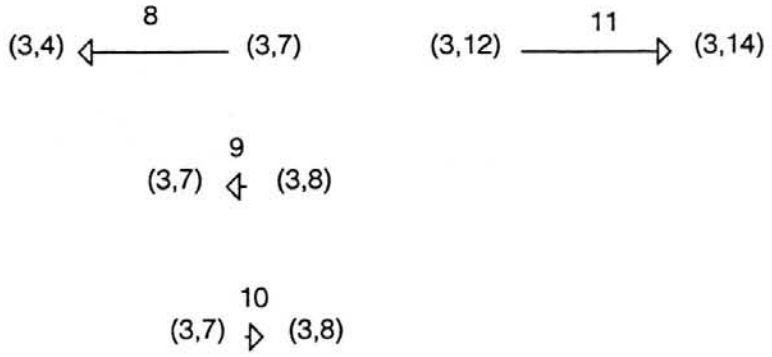


Figura 5.3 Conversão dos ramos da rede para coordenadas

C1 define que a figura de número 1 é um círculo

2 2 4 4 define o retângulo circunscrito ao círculo

0 0 12 5 definem o retângulo correspondente à área de interesse do círculo. Esta área envolve a anotação e a conexão associadas ao círculo.

R2 8 2 12 4 4 2 14 5 cj_aut <> {}

onde

R2 define que a figura 2 é um retângulo

8 2 12 2 coordenadas do retângulo

4 2 14 5 área de interesse

cj_aut <> {} anotação interna ao retângulo.

Temos ainda

C3 14 2 16 4 12 0 18 5
N4 0 0 6 2 0 0 6 4 autores (C_Aut)
N5 4 3 8 5 4 2 12 5 cj_aut_pres SUB cj_art
N6 12 3 14 5 8 2 16 5 {art}
N7 12 0 18 2 12 0 18 5 (C_Art)
X8 3 7 3 4

define uma seta apontando na direção de (3,4).

X9 3 8 3 7
X10 3 7 3 8
X11 3 12 3 14

As setas X8, X9 e X10 são sobrepostas para produzir o ramo que recebe a anotação N5. O resultado é mostrado na figura 15.

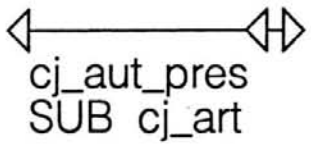


Figura 5.4 Resultado da composição de figuras básicas

5.6.2 Definição do universo de discurso

O exemplo trata de dois conjuntos:

C_Aut = { au1 , au2 }
C_Art = { ar1 , ar2 }

Colocando em termos da linguagem do sistema, temos as marcas definidas

M1 au1
M2 au2
M3 ar1
M4 ar2

5.6.2.1 Casos especiais

Existem pelo menos dois casos especiais importantes na definição de um universo de discurso: multiconjuntos e conjuntos de conjuntos. Estes casos serão analisados antes de continuar a definição da rede do exemplo.

O multiconjunto

C_Bat = { batata , batata , batata }

é representado pelo conjunto das marcas

M1 batata
M2 batata
M3 batata

O conjunto de conjuntos

C_Sal = { maionese , C_Bat }

é representado pelo conjunto das marcas

M4 maionese
M5 { batata , batata , batata }

O conjunto obtido por

C_Sal2 = { maionese , } C_Bat { }

é representado pelas marcas M1, M2, M3, M4.

5.6.3 Definição da dinâmica da rede

As marcas definidas para os lugares no exemplo são obtidas

L1 1 2
L2 3 4

Sendo L1 o lugar com anotação "autores (C_Aut)" e L2 o lugar com anotação "artigos (C_Art)". L1 pode conter as marcas M1 e M2, e L3 pode conter M3 e M4.

As regras de funcionamento dadas na figura 12 dizem que pelo menos um autor, e apenas 1 artigo são inseridos de cada vez. É possível inserir dois artigos de forma concorrente.

Temos então

A1 rec;ar1,{au1,au2} 2 1 0 1 1 0 2 3 0 3

significando "alteração (rec;ar1,{au1,au2}) definida pela conexão R2 está habilitada se M1 não está presente em L1, se M2 não está presente em L1 e se M3 não está presente em L3. Caso a conexão seja efetuada, fará com que M1 e M2 passem a estar presentes em L1 e M3 passe a estar presente em L3".

A2 rec;ar1,{au1,au2} 2 1 -1 1 1 0 2 3 0 3
A3 rec;ar1,{au1,au2} 2 1 0 1 1 -1 2 3 0 3
A4 rec;ar1,{au1,au2} 2 1 -1 1 1 -1 2 3 0 3

Significam em resumo que "se um autor já estiver presente coloca o outro autor e o artigo; se os doisautores estiverem presentes, coloca apenas o artigo".

Para ar2,{au1,au2} temos

A5 rec;ar2,{au1,au2} 2 1 0 1 1 0 2 3 0 4
A6 rec;ar2,{au1,au2} 2 1 -1 1 1 0 2 3 0 4
A7 rec;ar2,{au1,au2} 2 1 0 1 1 -1 2 3 0 4
A8 rec;ar2,{au1,au2} 2 1 -1 1 1 -1 2 3 0 4

Para ar1,au1 temos

A9 rec;ar1,au1 2 1 0 1 3 0 3
A10 rec;ar1,au1 2 1 -1 1 3 0 3

Para ar1,au2 temos

A11 rec;ar1,au2 2 1 0 1 3 0 4
A12 rec;ar1,au2 2 1 -1 1 3 0 4

Para ar2,au1 temos

A13 rec;ar2,au1 2 1 0 2 3 0 3
A14 rec;ar2,au1 2 1 -1 2 3 0 3

Para ar2,au2 temos

A15 rec;ar2,au2 2 1 0 2 3 0 4
A16 rec;ar2,au2 2 1 -1 2 3 0 4

Como o exemplo não define uma marcação inicial, a definição da rede está concluída.

6 SIMULADOR

6.1 Justificativa da estrutura do simulador

O simulador permite que o sistema encontre um caminho entre duas marcações, ou simplesmente execute um "passo" para a marcação seguinte. No protótipo do sistema o simulador utilizou uma implementação de banco de dados relacional, sobre o qual existia um núcleo de interpretação de comandos. Este sistema mostrou-se muito pouco eficiente em tempo de execução, embora ainda fosse uma alternativa interessante para a execução de comandos mais simples, tais como a execução de um "passo".

Optou-se por uma implementação otimizada do simulador, modificando-se a estrutura de dados da rede, e manteve-se o interpretador de comandos como um ponto de comunicação entre a interface do usuário e o sistema.

6.2 Notação interna da rede

A marcação da rede é guardada como uma matriz de bits, onde os índices são o número do lugar e o número da marca. O simulador admite até 32 marcas diferentes, e um número maior, mas limitado, de lugares.

No caso de marcação múltipla, admitida em alguns modelos de redes, cada marca ocupando um lugar é considerada (internamente) uma marca diferente, mesmo que tenha o mesmo nome de outra marca. Isto permite o tratamento de multiconjuntos, que são necessários para alguns modelos de redes.

A presença de uma marca em determinado lugar é determinada pelo estado do bit correspondente na matriz, como mostra a figura 16.

As alterações são relações entre marcas, formadas por conjuntos de tuplas do tipo <marca, atributo>. O atributo pode ser um de 4 valores de tipo de operação:

- testar se a marca está ausente, e depois torná-la presente;
- testar se a marca está presente, e depois torná-la ausente;

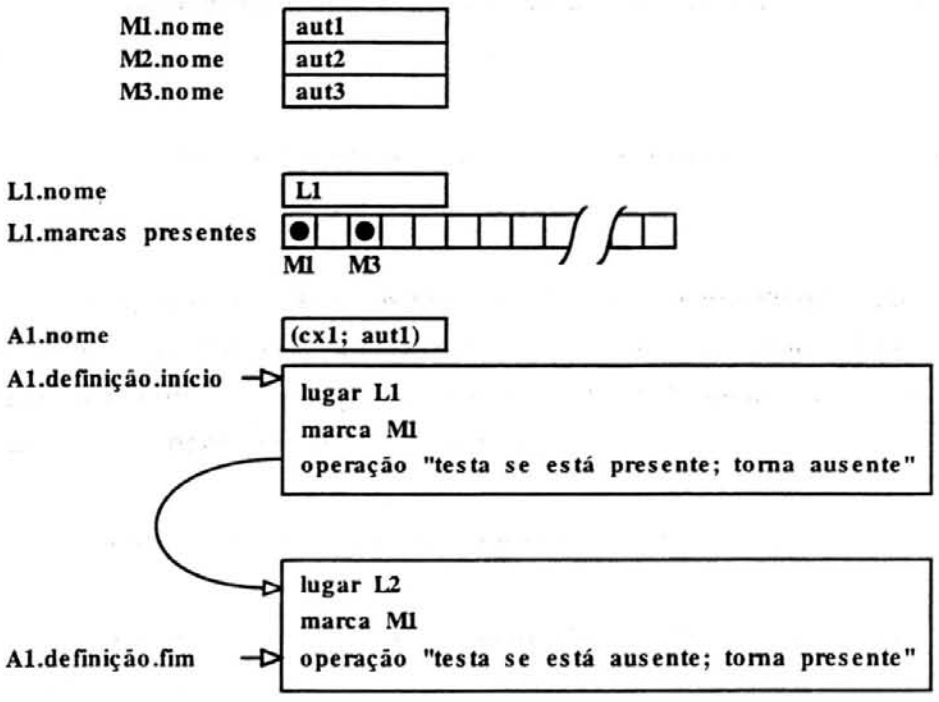


Figura 6.1 Representação de marcas, lugares e alterações

- testar se a marca está presente;
- testar se a marca está ausente.

As conexões mortas não possuem definição para o simulador.

6.3 Descrição do simulador

O simulador é o módulo responsável pela obtenção do caminho, se existente, entre duas marcações dadas. O simulador tem como dados de entrada:

- duas marcações, inicial e final;
- as alterações definidas.

A saída do simulador é uma lista de alterações sucessivas. O simulador procura o caminho mais curto entre as marcações.

O procedimento utilizado é o de simulação exaustiva, que termina quando um caminho foi encontrado ou quando ocorre uma condição de erro. Algumas condições de erro possíveis são a inexistência de alterações sucessoras, a volta à marcação de saída, ou a necessidade de um caminho muito longo. O tamanho máximo de um caminho é uma constante definida durante a implementação. O algoritmo é resumido nas figuras 17 e 18.

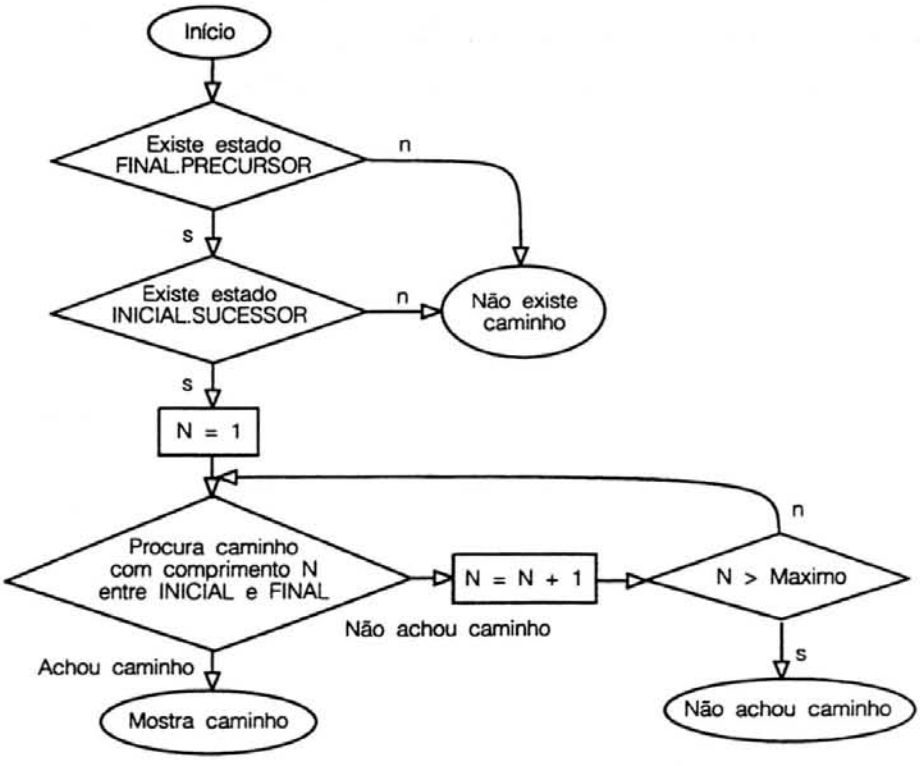


Figura 6.2 "Loop" externo da simulação exaustiva

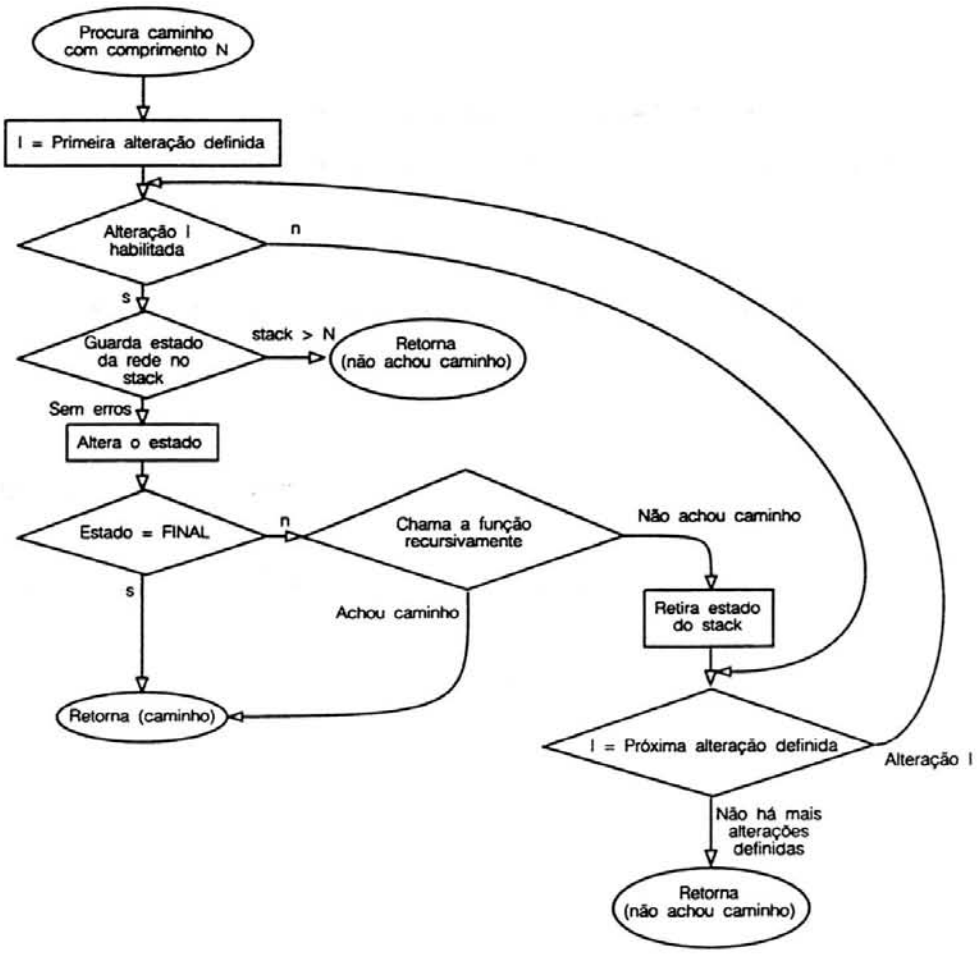


Figura 6.3 "Loop" interno da simulação exaustiva

6.4 Otimização do simulador

O simulador é o módulo do sistema que exigiu maior tempo de desenvolvimento, devido principalmente às otimizações que se fizeram necessárias. Estas otimizações foram realizadas principalmente a nível da representação de dados, a nível da implementação em linguagem Pascal. A representação como matriz de bits exigiu a redução do número máximo de marcas no universo de discurso para 32 marcas, permitindo a realização de testes sobre o estado da rede em apenas uma operação do tipo "longint". Por outro lado, a quantidade de alterações definidas, a quantidade de lugares e a quantidade de conexões não sofrem estas limitações, pois ficam armazenadas em listas.

Para reduzir o número de interações, foram utilizadas regras que reduzem o número de opções iniciais de transições e regras que evitam "loops". Existe um número máximo de passos para um caminho. Se o algoritmo não chega ao objetivo dentro do número máximo de passos, o usuário recebe uma mensagem e a procura é interrompida.

Devido a estas limitações, o algoritmo não pode ser utilizado para provar que não existe um caminho entre duas marcações. Existe uma exceção, no entanto, que ocorre quando a marcação final (objetivo) não tem marcação predecessora. Neste caso, o usuário é informado de que não existe um caminho.

As otimizações a nível de implementação constituíram-se na simplificação do endereçamento de variáveis e na utilização de pré-processamento sempre que possível.

As alterações efetuadas a nível de algoritmo e de implementação permitiram a viabilização da simulação sem a necessidade de utilizar linguagem assembler, o que colabora para a portabilidade do software.

Após a otimização, o ganho em velocidade do simulador foi significativo. Como o software é destinado a redes pequenas, o ganho em velocidade compensou a perda de capacidade de armazenamento.

7 FERRAMENTAS DE EDIÇÃO, VERIFICAÇÃO DE SINTAXE E TRADUÇÃO

7.1 Justificativa

A descrição de redes para o software implementado é feita através de uma linguagem considerada muito complicada para o professor. Embora o usuário final do sistema seja o aluno, o professor necessita de ferramentas que o auxiliem na preparação da aula.

Ao contrário das demais fases deste trabalho, em que se buscou a generalização, nesta fase foi escolhido um tipo específico de redes de alto nível, denominado redes compactas /HEU 90/. As redes compactas foram escolhidas por serem as utilizadas na disciplina "Modelagem de Sistemas com Redes de Petri", ministrado na Universidade Federal do Rio Grande do Sul.

Como o compilador é um módulo independente do editor gráfico, diversas linguagens de anotação poderão ser implementadas sobre o mesmo programa básico.

7.2 Especificação de um editor para o sistema implementado

O sistema implementado possui uma linguagem de descrição que contém todas as sentenças necessárias para a definição de uma rede de Petri de Alto Nível, a nível geométrico e a nível funcional. As redes que podem ser implementadas são limitadas apenas pela memória do sistema. A linguagem de descrição, no formato como é necessária para o sistema implementado, não é adequada para a utilização direta pelo aluno, pois exige um trabalho de preparação que deve ser executado manualmente.

7.2.1 Verificação de sintaxe

O sistema não permitirá a inclusão em seu banco de dados de:

- Linhas conectadas a mais de uma linha em uma extremidade;

- Linhas que formem figuras fechadas;

- Linhas que liguem retângulos com retângulos, ou elipses com elipses;

- Linhas que não tenham como início uma linha simples e como fim uma seta, ou como fim uma seta invertida e uma seta normal (símbolo de porta restauradora);

- Linhas que tenham interseção. Elipses são consideradas como ocupando a área do retângulo circunscrito. Linhas são consideradas de forma diferenciada das outras figuras;

- Inclusão de elementos que provoquem overflow de memória;

- Números das elipses repetidos; numeração não sequencial das elipses. O mesmo vale para os retângulos;

- Elipses ou retângulos sem a informação funcional correspondente, tal como é definida na linguagem de descrição de redes utilizada pelo sistema.

- Ramos que tratem de todas as entidades de um lugar.

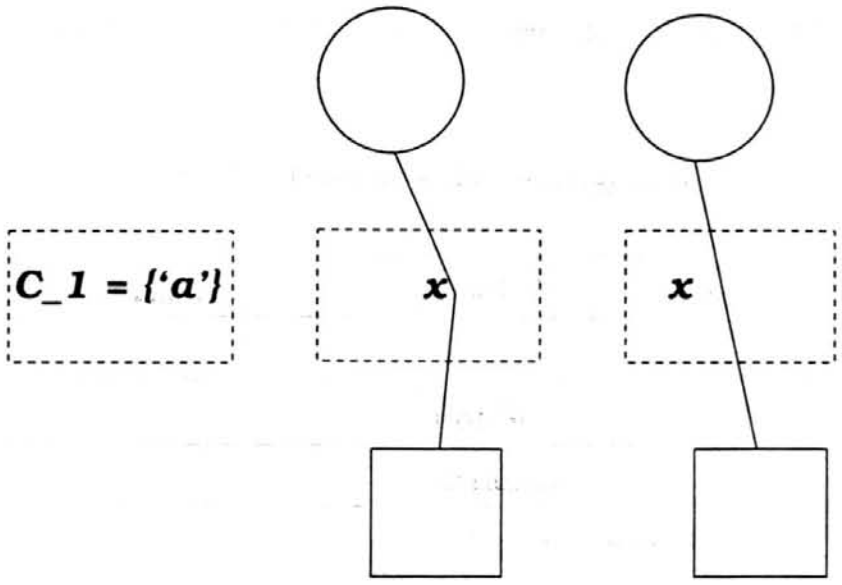
- Devido a limitações da especificação matemática utilizada para a tradução, o sistema não permitirá elementos de conjuntos com nomes repetidos, formando multiconjuntos. Os elementos de multiconjuntos podem, no entanto, ser numerados. Por exemplo: {m.1, m.2, m.3}

O compilador permite a existência de anotações sem texto.

Para o compilador reconhecer uma anotação de ramo é necessário que a anotação esteja sobre um nodo do ramo (lugar onde ocorre a união de duas linhas que compõem o ramo), como pode ser visto na figura 5.

Os erros causados por linhas totalmente desconectadas não são apontados pelo compilador.

O editor permite a inclusão de elementos que gerem erros de sintaxe. A verificação de sintaxe é feita somente pelo compilador.



a) anotação "global" b) anotação de ramo c) erro: anotação de ramo não conectada

Figura 7.1 Tipos de anotação

7.2.2 Interface gráfica

A interface gráfica utilizada é a mesma do sistema básico do simulador. Isto implica na utilização do mesmo sistema de tratamento de modificações da imagem, que provoca o apagamento da tela e novo desenho de toda imagem a cada modificação. Esta limitação simplifica a implementação do sistema, evitando a necessidade de programação específica para cada periférico. Com a evolução das bibliotecas gráficas disponíveis na linguagem utilizada, esta limitação poderá ser retirada. O mesmo sistema de coordenadas utilizado no sistema didático também foi mantido.

7.2.3 Interface com o usuário

O botão 2 do mouse chama o menu, que é análogo ao do sistema básico. Uma opção "edição" dá entrada no modo edição. Neste modo o menu tem comandos diferentes, seguidos de uma opção "fim da edição". O botão 1 do mouse é usado para seleção de posições na tela durante a edição.

Para modificar uma figura, o usuário deve apontar a figura e apertar o botão 1 do mouse. Após selecionar uma figura, o usuário pode:

- Chamar o menu, pressionando o botão 2 do mouse. O menu de edição é descrito a seguir.

- Apontar outro lugar da tela e pressionar o botão 1 do mouse novamente. O sistema mudará a posição do vértice do retângulo circunscrito à figura selecionada, que encontrar-se mais próximo à posição do mouse. A forma da figura é então alterada. A modificação de uma figura não afeta as outras figuras.

- Pressionar a tecla "delete", eliminando a figura.

Qualquer modificação na geometria da rede será seguida de uma verificação das fronteiras do desenho, permitindo que o editor adapte seu sistema de coordenadas ao desenho. Desta forma, o sistema continuará sempre utilizando a precisão máxima disponível no dispositivo gráfico utilizado.

A descrição do funcionamento é toda feita em linguagem de alto nível. O usuário não precisa ter acesso à rede subjacente. A linguagem de anotação é bastante similar à descrita em /HEU 90/. A entrada de texto em uma figura é feita selecionando-se a figura e chamando a opção "altera texto" do menu.

7.2.4 Menu do editor

O editor é acessado através da opção "editar" do menu principal. Dentro do editor, existe um outro menu, específico para as funções do editor (figura 4).

O menu do editor possui as seguintes opções:

- Compila: realiza a verificação de sintaxe da rede e gera a rede subjacente na linguagem própria do software didático. O resultado não é gravado em disco.

- Opções Elipse, Retângulo, Fato, Anotação, Seta, Linha: inserem novas figuras na rede. A nova entidade poderá ser modificada posteriormente, utilizando-se os comandos de edição já descritos.

- Altera texto: permite a modificação do texto relacionado a uma figura.

- Save: grava a rede em disco. Se a rede foi compilada, a descrição do funcionamento também é gravada. Como o usuário pode desejar a gravação de um trabalho que ainda não está completo, a gravação não exige que a rede esteja correta.

- Fim da Edição: volta ao modo normal do sistema didático.

7.2.5 Diferenças entre a linguagem de anotação implementada e a descrita em /HEU 90/

A implementação do compilador utilizando a calculadora lógica permitiu a especificação de uma linguagem bastante similar à desejada. Algumas diferenças, no entanto, ainda permaneceram:

- Os ramos não podem conter fórmulas, apenas termos. Os termos devem ter o tipo determinado pelo lugar.

- Algumas operações foram implementadas com nomes diferentes dos sugeridos: NOT em vez de nao, AND em vez de e, OR em vez de ou, T em vez de Verdadeiro, F em vez de Falso e DIFERENTE em vez de ≠. As demais operações foram implementadas com o mesmo nome e sintaxe, mas utilizando letras maiúsculas e sem o sublinhado. Somente operações lógicas (não aritméticas) são permitidas nesta versão.

- Não existem símbolos de relação definidos pelo usuário.

- O termo { <var> | <fórmula> } foi implementado sem as chaves.

- É necessário um espaço entre cada elemento de sintaxe, incluindo parênteses, vírgulas e chaves.

- Ramos sem variáveis livres habilitam a passagem de qualquer marca. A especificação em /HEU 90/ não fala sobre o que fazer neste caso.

7.3 Implementação do tradutor de redes de alto nível

A opção "Compila" do menu do editor executa uma rotina de verificação de sintaxe e de tradução do desenho da rede de alto nível para a linguagem interna do simulador didático.

A rotina de tradução utiliza uma calculadora lógica, implementada separadamente, que é responsável pela maior parte da interpretação da linguagem de anotação da rede.

A "compilação" é realizada em modo texto, para acelerar a apresentação de mensagens.

7.3.1 Mensagens de erro do compilador

a) Ramo toca mais de um Retangulo ou Conexao morta

b) Ramo toca mais de uma figura alem da Elipse

c) Ramo toca mais de de uma anotacao

d) Ramo toca mais de uma figura ou linha

e) Ramo invalido

- ramo não define um tipo de porta válida (entrada, saída, restauradora)

f) Formula de conexao errada

g) A variavel livre ... tem o mesmo nome de uma marca

h) símbolo de atribuição não encontrado

- definições globais (dentro de anotações) devem ter a forma:

<var> = (<definição>)

i) o valor do simbolo esta' mal delimitado

- falta parênteses em uma definição global.

j) Elipse toca Conexao morta

k) Elipse toca Elipse

l) Elipse toca Retangulo

m) Elipse toca mais de de uma anotacao

n) A marca ... nao faz parte do universo de discurso

o) A marca ... nao pode estar neste lugar

p) Retangulo toca Conexao morta

q) Retangulo toca Elipse

r) Retangulo toca Retangulo

s) Retangulo toca mais de de uma anotacao

t) Figura muito pequena, nao pode determinar a direcao

- o programa não pode determinar a direção de uma seta.

u) Conexao sem ramos

v) Foram encontrados erros. A compilacao foi cancelada.

- Esta mensagem é mostrada sempre que ocorrem um dos erros anteriores.

7.4 Implementação de uma calculadora lógica para a linguagem de anotação

Para permitir a tradução automática de descrições de redes de alto nível para a descrição utilizada pelo sistema didático, foi realizada a implementação de uma calculadora, capaz de interpretar os termos da linguagem de anotação descrita em /HEU 90/.

A calculadora foi implementada inicialmente com notação rpn, como um programa independente, sendo depois transformada para notação infixada e integrada ao editor gráfico do simulador didático.

7.4.1 A linguagem de anotação e a calculadora

A linguagem de anotação destina-se a falar sobre as entidades do modelo em questão. Uma entidade é um representante abstrato dos objetos da realidade modelada. Tuplas formadas por entidades são tratados como sendo também entidades. Na calculadora implementada, tuplas recebem o nome de conjunto ordenado, e são denotadas por < *entidade entidade ...* >.

Para falar sobre entidades, a linguagem de anotação possui termos, que são expressões que designam uma entidade, e fórmulas, que são expressões que designam um valor lógico (verdadeiro ou falso). Na calculadora implementada, termos e fórmulas também são entidades.

O domínio de um lugar, na linguagem de anotação, corresponde na calculadora a uma entidade.

A calculadora não possui a capacidade de ligar (valorar) variáveis. Esta capacidade deve pertencer ao sistema que irá usar a calculadora. Esta retornará uma entidade para quem fez a consulta. Esta entidade pode ser um conjunto, um conjunto ordenado, um valor lógico ou um identificador (constante que designa uma entidade do universo de discurso). A calculadora não reconhece também símbolos de função e símbolos de relação. Estes símbolos devem ser pré-processados e seu significado entregue à calculadora. As fórmulas *PARATODO var fórmula*, *EXISTE var fórmula*, *{ var | fórmula }*, que dependem de valoração de variáveis, também devem ser implementadas pelo sistema em nível mais alto, utilizando as funções básicas fornecidas pela calculadora.

A linguagem de anotação é uma linguagem de lógica de predicados de primeira ordem, e que contém símbolos para o tratamento de conjuntos.

7.4.2 Sintaxe da calculadora rpn

A calculadora rpn /GER 87/ (figura 20) é uma rotina que recebe como dado de entrada uma pilha (stack) contendo entidades, e retorna a pilha processada.

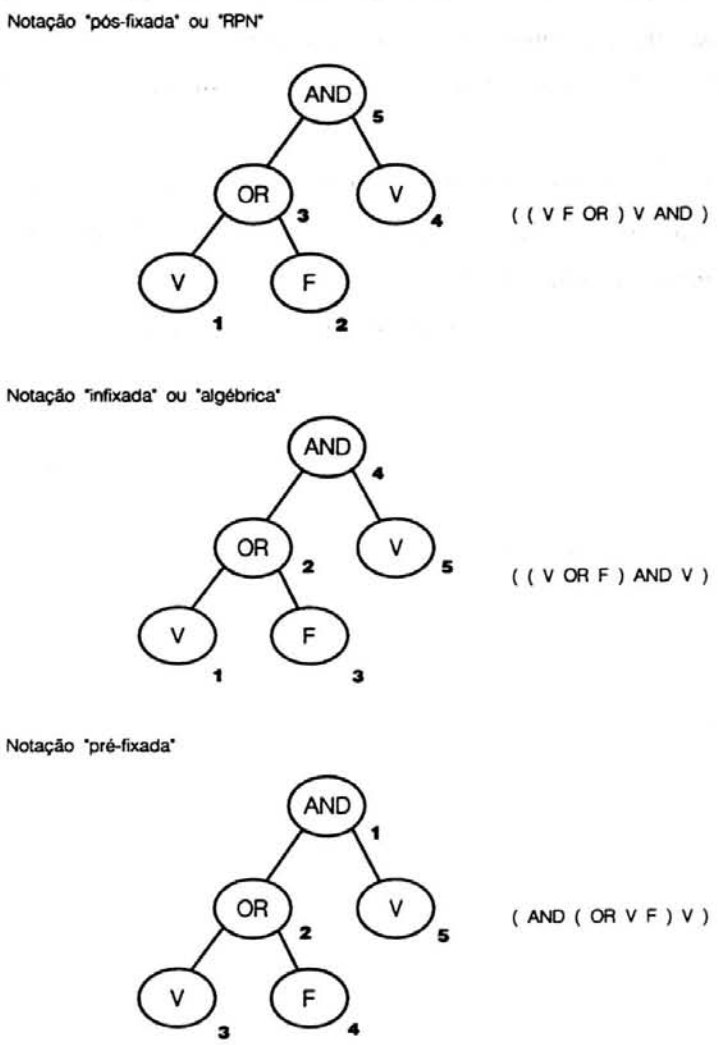


Figura 7.2 Formas de leitura das operações da calculadora

Os termos apresentados na tabela 3 são alguns exemplos da linguagem da calculadora.

Tabela 7.1 Exemplos da linguagem rpn

Conteúdo da Entrada	Conteúdo da Saída
TRUE	TRUE
FALSE	FALSE
entidade	entidade
entidade DUP	entidade entidade
{ entidade entidade ... }	{ entidade entidade ... }
< entidade entidade ... >	< entidade entidade ... >
{ entidade entidade ... } entidade ELEMENTO	TRUE ou FALSE
{ entidade entidade ... } { entidade entidade ... } SUBCONJUNTO	TRUE ou FALSE
{ entidade entidade ... } { entidade entidade ... } UNIAO, INTERSECAO, X	{ entidade entidade ... } Obs: As operações tratam de multiconjuntos.
{ entidade entidade ... } POTENCIA	{ subconjunto subconjunto ... }
entidade entidade IGUAL, DIFERENTE	TRUE ou FALSE
função, TRUE ou FALSE função, TRUE ou FALSE E, OU, IMPLICA	TRUE ou FALSE
função, TRUE ou FALSE NAO	TRUE ou FALSE

7.4.3 Implementação da calculadora rpn

A calculadora foi implementada em linguagem Pascal orientada a objetos, a fim de permitir a posterior integração com o sistema didático.

O stack da calculadora foi limitado nos testes a 20k bytes, o que pareceu suficiente para todas as situações encontradas em redes de alto nível descritas em /HEU 90/.

A calculadora foi definida como um método, que define ao mesmo tempo as entidades e as operações principais:

PushStr (string); PopStr (string) - usados para modificar o conteúdo de uma entidade.

PushEntidade (entidade), PopEntidade (entidade) - usados para o transporte de entidades entre pilhas.

Eval - usado para simplificar uma entidade do tipo fórmula ou termo.

Outros métodos foram implementados para facilitar o trabalho de testagem de programas: Print (mostra o conteúdo da pilha), IsEmpty (retorna true se a pilha está vazia).

7.4.4 Exemplo de utilização da calculadora por um simulador de redes de Petri

Uma rede de Petri possui um lugar chamado "dados do empregado", que tem como marcas definidas "(C_Nome X C_Salário)". C_Nome é definido como {'antonio', 'joao'} e C_Salário é definido como { 10.000, 20.000 }. O lugar tem marcação atual { <'antonio', 20.000 > }.

Para valorar um ramo com anotação "<n,s>", o sistema envia à calculadora a mensagem:

PushStr ('{ <'antonio', 20.000 > }') conteúdo do lugar

PopStr (st) retira o sinal '{', "abrindo" o conjunto

PopEntidade (EntAux) retira a tupla <'antonio', 20.000 >

EntAux.PopStr (st) retira o sinal '<', "abrindo" a tupla

EntAux.PopStr (n) valora a variável n

EntAux.PopStr (s) valora a variável s

PopEntidade (EntAux) verifica se há mais tuplas.

EntAux.PopStr (st) retira o sinal '}', indicando fim das entidades presentes.

- Tendo obtido o valor de n, o sistema pode verificar se uma variável n proveniente de outro ramo contém o mesmo valor, através de:

PushStr (' 'antonio' 'antonio' IGUAL ')

Eval

PopStr (st)

- st conterá o valor 'TRUE'

7.4.5 Transformação da calculadora rpn em notação infixada

Após a realização dos testes com a calculadora rpn, a função Eval foi alterada permitindo a operação em modo infixado, que é mais parecida com a linguagem de anotação descrita em /HEU 90/.

A sintaxe recebeu as seguintes modificações:

- Parênteses são permitidos;

Tabela 7.2 Termos implementados com a função "tal que"

Notação da linguagem de anotação	Fórmula utilizada para execução interna
Conj1 UNIAO Conj2	(x | x ELEM Conj1 OR x ELEM Conj2)
Conj1 INTERSECAO Con j2	(x | x ELEM Conj1 AND x ELEM Conj2)
EXISTE var1 fórmula	((x | fórmula) DIFERENTE { })
PARATODO var1 fórmula	((x | NOT (fórmula)) = { })

- Os operadores têm ordem de prioridade;

- O operador Dup deixa de existir como comando da calculadora. A estrutura dos comandos básicos continua sendo baseada em pilhas.

Para valorar um ramo com anotação "<n,s>" como no exemplo anterior, o sistema, tendo obtido o valor de n, pode verificar se uma variável n proveniente de outro ramo contém o mesmo valor, através de:

PushStr (' 'antonio' = 'antonio' ')

Eval

PopStr (st)

- st conterá o valor 'TRUE'

Os nomes de operações, tal como a palavra IGUAL ou o símbolo "=" são opção do implementador, podendo inclusive fazer parte de um dicionário de dados do sistema. Isto possibilita a existência de versões em português ou em inglês da linguagem.

7.4.6 Operações com conjuntos

A calculadora possui o comando "tal que", com a seguinte sintaxe: <variável independente> | <função lógica>. Este comando cria um conjunto em que os elementos são os valores verdade para a função.

Utilizando a função "tal que", foram implementados os termos da linguagem de anotação listados na tabela 4.

É possível implementar as relações "SUB" ("é contido") e a igualdade de conjuntos utilizando a notação acima, mas optou-se por implementar estes comandos utilizando a versão rpn da calculadora, para maior otimização. Para efeito de ilustração, são dadas na tabela 5 as fórmulas que seriam utilizadas para estes comandos.

A calculadora utiliza um mecanismo de "execução inversa" para determinar as possíveis respostas para a função "tal que". As respostas são testadas na fórmula através da substituição da variável livre, antes de retornar ao programa principal da calculadora.

7.4.7 Problemas encontrados na linguagem de anotação

a) Operação "produto de conjuntos"

A expressão (x1 X x2) não pôde ser implementada diretamente como operação binária, pois o resultado não pôde ser compatibilizado com o definido em /HEU 90/.

Tabela 7.3 Implementação de SUB e "=" com "tal que"

Notação da linguagem de anotação	Fórmula utilizada para execução interna
Conj1 SUB Conj2	(x | x ELEM Conj1 IMPL x ELEM Conj2)
Conj1 = Conj2	(x | Conj1 SUB Conj2 AND Conj2 SUB Conj1)

Para uma expressão ({ 1 , 2 } X { 4 , 5 } X { 6 , 7 }), o sistema teria como resposta

{ <<1,4>,6>, <<1,4>,7>, ... }

quando o resultado esperado seria

{ <1,4,6>, <1,4,7>, ... }

A fim de manter a compatibilidade com a definição da linguagem de anotação, a operação produto de conjuntos foi definida da seguinte forma:

Tabela 7.4 Operação "produto de conjuntos"

Notação da linguagem de anotação	Fórmula utilizada para execução interna
Conj1 X Conj2	(<x1, x2> | x1 ELEM Conj1 AND x2 ELEM Conj2)
Conj1 X Conj2 X ... X $Conj_n$	(<x1, x2, ..., x_n> | x1 ELEM Conj1 AND x2 ELEM Conj2 AND ... AND x_n ELEM $Conj_n$)

b) Listas formadas por variáveis livres

Utilizando a linguagem de anotação como linguagem básica do simulador, poderia ser utilizada uma expressão como a seguinte para determinar se uma alteração com marcas de entrada <n,s> e n está habilitada:

EXISTE <n,s> (<n,s> ELEM C_Lugar1 AND n ELEM C_Lugar2)

Esta expressão, no entanto, não faz parte da linguagem de anotação, pois <n,s> não é uma variável livre.

Como existem exemplos na literatura em que a notação acima foi utilizada, foi acrescentada à calculadora a sintaxe:

<lista> | <função lógica>

A lista pode conter variáveis livres ou constantes. Por exemplo:

<x, y, 3> | x = y AND y = 2

c) Multiconjuntos

A necessidade de trabalhar com multiconjuntos torna o resultado de algumas operações duvidoso. Por exemplo, a união de conjuntos seguida de uma intersecção de conjuntos poderia eliminar uma quantidade errada de elementos repetidos.

Este problema ainda não foi resolvido a nível da calculadora. A linguagem básica do simulador admite que entidades diferentes tenham o mesmo nome, dando ao usuário a impressão de que se trata da mesma entidade. Este procedimento não pode, aparentemente, ser estendido ao nível mais alto da rede.

7.4.8 Observações

Tanto a calculadora lógica rpn como a de notação infixada foram implementadas, mas apenas a de notação infixada possui a função "talque". A calculadora infixada é a que melhor se adaptou às necessidades do software, pois suas sentenças são mais próximas às da linguagem de anotação desejada.

A calculadora lógica é um bloco básico para a construção do interpretador para redes de Petri de alto nível, contendo as operações básicas com conjuntos e listas. Ela deve ser incorporada a um editor gráfico, para produzir redes no formato adequado ao sistema didático já implementado, ou tornar-se parte do próprio simulador, aumentando o nível da linguagem de descrição de redes de Petri.

A utilização da linguagem de anotação como linguagem básica do simulador é possível, embora a velocidade do sistema venha a tornar inviável a utilização de comandos do tipo "achar caminho".

Utilizando a linguagem de anotação como linguagem básica do simulador, poderia ser utilizada uma expressão como a seguinte para determinar se uma alteração com marcas de entrada <n,s> e n está habilitada:

EXISTE <n,s> (<n,s> ELEM C_Lugar1 AND n ELEM C_Lugar2)

Quanto às entidades reconhecidas pela calculadora como funções, poderiam também ser implementadas a inclusão e exclusão de elementos em conjuntos e a subtração de conjuntos.

Também poderiam fazer parte das funções a ligação de variáveis e funções de atribuição, e a modificação de variáveis globais (tais como a marcação da rede) e substituição de identificadores (tais como nomes de conjuntos). Estas implementações adicionais simplificariam o trabalho do sistema em que a calculadora se insere, permitindo uma capacidade de expressão maior.

Para complementar as funções lógicas e de conjuntos, a calculadora também poderia realizar operações matemáticas aritméticas, sem que fosse necessário alterar sua estrutura básica. Estas implementações exigiriam um cuidado especial com a função "talque", já que o seu modo de funcionamento, baseado na obtenção de hipóteses sobre sentenças, não é um método adequado para a solução de equações aritméticas.

7.5 Sintaxe da linguagem de alto nível implementada

A sintaxe da linguagem implementada no compilador é apresentada na tabela 7. Esta tabela deve ser comparada à tabela 1 que serviu de base para sua definição. As figuras 19 e 21 auxiliam o entendimento de sintaxe.

Tabela 7.5 Sintaxe da linguagem de alto nível

<*conexão*> ::=	conexão | conexão morta
<*porta*> ::=	ramo alterador de entrada | ramo alterador de saída | ramo restaurador de entrada | ramo restaurador de saída
ramo alterador de entrada ::=	borda de *conexão*, seta apontada para *conexão*, [linha, ...], borda de lugar
ramo alterador de saída ::=	borda de *conexão*, [linha, ...], seta apontada para lugar, borda de lugar
ramo restaurador de entrada ::=	borda de *conexão*, seta apontada para *conexão*, seta apontada para lugar, linha, [linha, ...], borda de lugar
ramo restaurador de saída ::=	borda de *conexão*, linha, [linha, ...], seta apontada para *conexão*, seta apontada para lugar, borda de lugar

anotação global ::= *nome de conjunto = conjunto*

anotação de ramo ::= *expressão que define marca* e toca um nodo de uma *porta e* pertence ao conjunto definido por *tipo de marca do lugar* referente a *porta* | *expressão que define lista e* toca um nodo de uma *porta e* cada item da *lista* pertence ao conjunto definido por *tipo de marca do lugar* referente a *porta*

tipo de marca do lugar ::= [(*expressão que define conjunto*)] [comentário que define o significado do lugar para o usuário] *e* toca um lugar

fórmula de conexão ::= expressão lógica *e* está dentro de uma *conexão*

nome de conexão ::= constante *e* toca *conexão*

marcação inicial ::= *marca* [, *marca* ...] | *nome de conjunto*

marca ::= constante |
lista |
conjunto |
nome de conjunto

lista ::= < *marca*, ... >

conjunto ::= { *marca*, ... }

nome de conjunto ::= C_ Constante

expressão que define conjunto ::= *nome de conjunto* |
{ *expressão que define marca*, ... } *expressão que define conjunto* UNIAO *expressão que define conjunto* |
expressão que define conjunto INTERSECAO *expressão que define conjunto* |
POTENCIA *expressão que define conjunto* |
expressão que define conjunto X *expressão que define conjunto* [X *expressão que define conjunto* ...] |
variável | *expressão lógica* |
< variável, ... > | *expressão lógica* |
(*expressão que define conjunto*)

expressão que define marca ::= marca |
expressão que define conjunto |
expressão que define lista |
(expressão que define marca)

expressão que define lista ::= < expressão que define marca, ... >

expressão lógica ::= T |
F |
expressão que define marca ELEM *expressão que define conjunto* |
expressão que define conjunto SUB *expressão que define conjunto* |
expressão que define marca = *expressão que define marca* |
expressão que define marca DIFERENTE *expressão que define marca* |
expressão que define conjunto = *expressão que define conjunto* |
expressão que define conjunto DIFERENTE *expressão que define conjunto* |
expressão que define lista = *expressão que define lista* |
expressão que define lista DIFERENTE *expressão que define lista* |
NOT *expressão lógica* |
expressão lógica OR *expressão lógica* |
expressão lógica AND *expressão lógica* |
expressão lógica IMPL *expressão lógica* |
EXISTE variável *expressão lógica* |
PARATODO variável *expressão lógica* |
EXISTE < variável, ... > *expressão lógica* |
PARATODO < variável, ... > *expressão lógica* |
(*expressão lógica*)

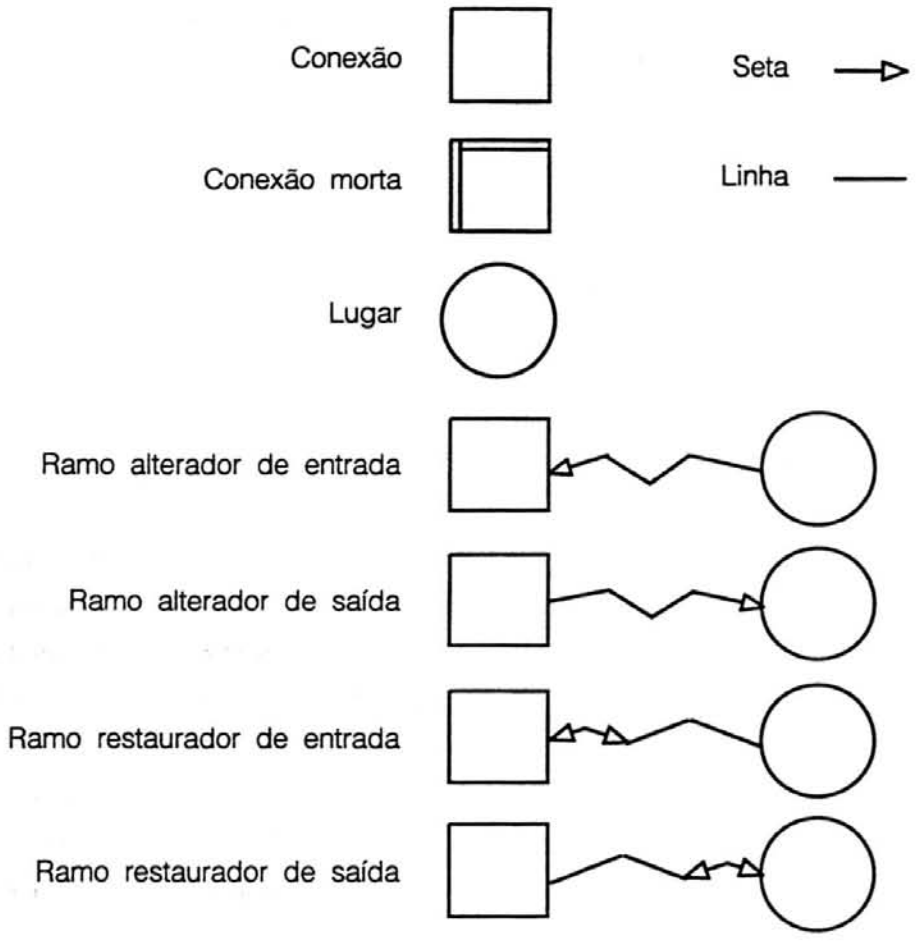


Figura 7.3 Elementos gráficos da linguagem implementada

8 CONCLUSÕES E RECOMENDAÇÕES

O sistema desenvolvido permite a demonstração das características gráficas e da semântica de alguns modelos tradicionais de redes de Petri, tais como redes condição/evento, redes predicado-transição e redes coloridas.

Algumas redes já foram implementadas no sistema, permitindo sua demonstração para grupos de alunos de forma experimental.

A linguagem de descrição vinha sendo a principal dificuldade para a utilização do sistema em situações reais de sala de aula. Por isso, foi desenvolvido um editor gráfico e tradutor de redes de alto nível para a linguagem de descrição, permitindo a edição de redes na própria sala de aula.

Foi também pesquisada a possibilidade de implementar-se uma interface entre um editor para diagramas comercial e a linguagem de definição utilizada pelo sistema implementado. Foram realizadas experiências sobre o software Design, em estação de trabalho Sun.

8.1 Problemas encontrados na implementação atual

O simulador não foi planejado para encontrar um caminho entre duas marcações em redes que contenham conexões mortas.

A opção "Executa Passo" do menu apresenta um problema de implementação, que ocorre quando duas alterações têm conflito através de uma conexão morta. Um conflito deste tipo só ocorre quando as alterações que compõem o passo acontecem, e o sistema permite então a seleção de ambas alterações. Quando o passo é executado pode ocorrer o conflito, e a marcação sucessora não estará correta. Este problema poderá ser resolvido em uma implementação futura, através do seguinte algoritmo:

1. i := 0
2. Executa passo
3. Se i > Limite
   Fim (conflito não resolvido)
3. Se alguma alteração do passo não foi executada
   i := i + 1
   Volta para 2

8.2 Especificações desejáveis em novas implementações

O sistema modela a forma gráfica das Redes, não podendo ser representada a forma matricial. Seria desejável em implementações posteriores que o usuário tivesse acesso a ambas as formas.

O sistema não permite a representação de símbolos matemáticos. Seria interessante a implementação do software em um ambiente gráfico que possuísse tais caracteres. Tal ambiente não deverá, no entanto, limitar o hardware a computadores não portáteis, pois isto limitaria a disponibilidade do sistema. Outros pontos a serem melhorados com a introdução de um ambiente gráfico de melhor qualidade são: evitar que o software tenha que redesenhar a tela cada vez que uma alteração seja feita na apresentação; fornecer retorno para o usuário sobre a função escolhida com o mouse, que permita verificar se é efetivamente a função desejada; permitir a mudança de posição das janelas a gosto do usuário; visualização do estado do sistema (por exemplo: opção "seleciona" e "executa alterações" do menu); utilização de cores.

8.3 Redes implementadas com o sistema proposto

O anexo A apresenta à consideração do leitor algumas redes que exemplificam a operacionalização do sistema proposto.

As redes A.1, A.2 e A.3 foram editadas e compiladas pelo próprio sistema. A rede A.4 foi descrita manualmente utilizando a linguagem de baixo nível, a fim de permitir a utilização de um multiconjunto.

ANEXO A EXEMPLOS DE REDES IMPLEMENTADAS

A.1 Rede MERCADO.DAT

Implementa uma rede compacta que representa um modelo de "mercado de trabalho", onde é possível a criação de vagas e a entrada de pessoas no mercado de trabalho. A realização de um contrato é representada pela criação de tuplas do tipo <pessoa, vaga>.

A.2 Rede SEMAFORO.DAT

Implementa um modelo de semáforo, onde os carros dependem do estado de um semáforo para poderem passar de um lugar para outro.

A.3 Rede MORTA.DAT

Exemplo simples de rede com conexões mortas.

A.4 Rede MULTI.DAT

Utiliza o mesmo objeto da rede SEMAFORO para demonstrar uma modelagem com multiconjuntos. Nesta redes todos os carros são representados por "a".

A.1 Rede MERCADO.DAT

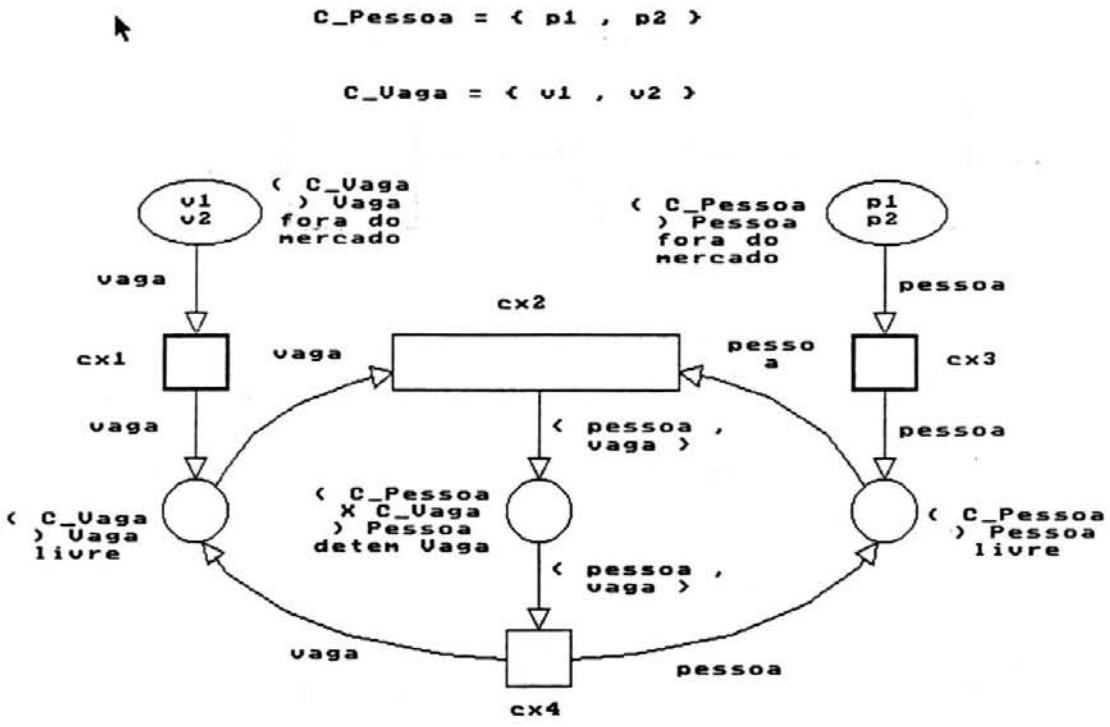


Figura A.1 Rede MERCADO.DAT

```
; Arquivo gerado pelo simulador didatico de redes
; de Petri de alto nivel.
;
; Flavio Soibelmann Glock 1990/1991
;
; Nome do arquivo: mercado.dat
;
; Geometria da rede
E1 2 4 21 14  2 4 47 28  v1 v2
E2 6 51 16 61  -1 44 23 68
E3 60 51 70 61  36 37 87 75
```

E4 114 51 124 61 107 44 131 68
E5 110 4 129 14 87 2 142 28 p1 p2
R6 6 28 16 37 -12 14 16 51
R7 42 28 87 37 21 19 110 51
R8 114 28 124 37 114 14 142 51
R9 60 75 70 84 27 61 102 92
N10 21 2 47 17 21 2 47 17 (C_Vaga) Vaga fora do mercado
N11 -12 16 15 24 -31 10 34 30 vaga
N12 -12 28 6 37 -12 28 6 37 cx1
N13 -12 39 13 47 -30 33 31 53 vaga
N14 -20 51 6 70 -38 38 24 83 (C_Vaga) Vaga livre
N15 53 19 73 28 39 13 87 34 cx2
N16 19 24 39 39 5 14 53 50 vaga
N17 92 25 112 38 78 16 126 47 pessoa
N18 63 39 100 50 37 31 126 58 pessoa , vaga
N19 28 48 60 68 6 34 82 82 (C_Pessoa X C_Vaga) Pessoa detem Vaga
N20 64 62 99 73 40 54 124 81 pessoa , vaga
N21 17 72 47 87 -4 62 68 98 vaga
N22 53 84 77 92 53 84 77 92 cx4
N23 78 77 113 87 54 70 138 94 pessoa
N24 77 3 110 22 54 -10 133 35 (C_Pessoa) Pessoa fora do mercado
N25 118 17 142 25 101 11 159 31 pessoa
N26 124 28 142 37 124 28 142 37 cx3
N27 118 39 142 48 101 33 159 54 pessoa
N28 124 51 156 69 102 38 178 82 (C_Pessoa) Pessoa livre
X29 11 20 11 28
X30 11 43 11 51
X31 119 21 119 28
X32 119 42 119 51
X33 35 36 42 34
X34 92 35 87 34
X35 115 65 118 61
X36 16 66 12 61
X37 65 43 65 51
X38 65 67 65 75
Z39 14 52 16 49

```
Z40 16 49 20 44
Z41 20 44 27 39
Z42 27 39 35 36
Z43 116 52 114 49
Z44 114 49 110 44
Z45 110 44 101 38
Z46 101 38 92 35
Z47 25 72 16 66
Z48 35 76 25 72
Z49 50 79 35 76
Z50 60 80 50 79
Z51 70 80 80 79
Z52 80 79 95 76
Z53 95 76 107 71
Z54 107 71 115 65
Z55 11 14 11 20
Z56 11 37 11 43
Z57 119 14 119 21
Z58 119 37 119 42
Z59 65 37 65 44
Z60 65 61 65 68
N61 24 -15 101 -4  -30 -23 155 4  C_Vaga = { v1 , v2 }
N62 20 -26 101 -17  -37 -32 158 -11  C_Pessoa = { p1 , p2 }
;
; Universo de discurso
M1 v1
M2 v2
M3 p1
M4  p1 , v1
M5  p1 , v2
M6 p2
M7  p2 , v1
M8  p2 , v2
;
; Dominio dos lugares
L1 1
```

```
L1 2
L2 1
L2 2
L3 4
L3 5
L3 7
L3 8
L4 3
L4 6
L5 3
L5 6
;
; Alteracoes definidas
A1 6 cx1;v1  1 1 0  2 0 1
A2 6 cx1;v2  1 2 0  2 0 2
A3 7 cx2;p1,v1  2 1 0  4 3 0  3 0 4
A4 7 cx2;p2,v1  2 1 0  4 6 0  3 0 7
A5 7 cx2;p1,v2  2 2 0  4 3 0  3 0 5
A6 7 cx2;p2,v2  2 2 0  4 6 0  3 0 8
A7 8 cx3;p1  5 3 0  4 0 3
A8 8 cx3;p2  5 6 0  4 0 6
A9 9 cx4;p1,v1  3 4 0  2 0 1  4 0 3
A10 9 cx4;p1,v2  3 5 0  2 0 2  4 0 3
A11 9 cx4;p2,v1  3 7 0  2 0 1  4 0 6
A12 9 cx4;p2,v2  3 8 0  2 0 2  4 0 6
;
; Marcacao inicial
I1 1
I1 2
I5 3
I5 6
;
; fim do arquivo
```

A.2 Rede SEMAFORO.DAT

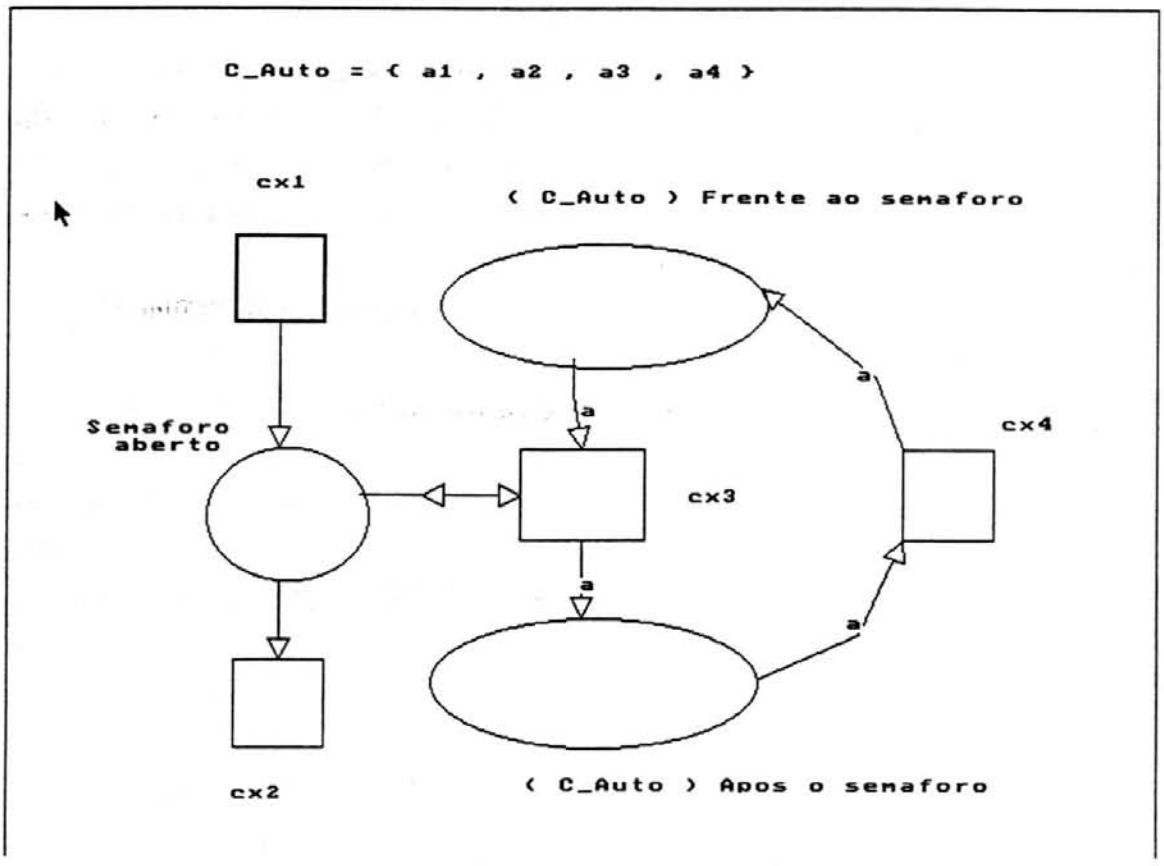


Figura A.2 Rede SEMAFORO.DAT

```
; Arquivo gerado pelo simulador didatico de redes
; de Petri de alto nivel.
;
; Flavio Soibelmann Glock 1990/1991
;
; Nome do arquivo: semaforo.dat
;
; Geometria da rede
E1 67 27 85 42  54 17 98 53
E2 93 4 130 18  67 -6 156 28
E3 92 46 129 61  66 36 155 72
R4 70 3 80 13  70 3 95 27
```

```
R5 70 51 80 61  70 37 95 61
R6 102 27 116 37  92 20 126 44
R7 145 27 155 37  129 14 170 61
N8 63 -8 88 3  46 -16 106 11  cx1
N9 52 17 72 35  38 4 86 48  Semaforo aberto
N10 58 61 88 72  37 53 109 80  cx2
N11 93 -7 166 5  42 -15 217 13  ( C_Auto ) Frente ao semaforo
N12 115 27 133 38  102 19 146 46  cx3
N13 93 59 165 72  43 50 215 81  ( C_Auto ) Apos o semaforo
N14 153 17 165 32  145 7 173 43  cx4
X15 75 13 75 27
X16 75 42 75 51
X17 108 21 109 27
X18 109 41 109 46
X19 142 19 129 9
N20 54 -21 144 -9  -9 -29 207 -1  C_Auto = { a1 , a2 , a3 , a4 }
X21 140 48 145 37
X22 95 32 102 32
X23 95 32 91 32
Z24 84 32 91 32
N25 138 16 144 22  136 14 146 23  a
N26 137 44 143 50  135 42 144 51  a
Z27 142 19 145 27
Z28 129 53 140 48
Z29 108 21 108 17
N30 107 20 113 26  106 18 115 27  a
Z31 109 37 109 42
N32 107 40 113 45  103 37 117 49  a
;
; Universo de discurso
M1 ~
M2 a1
M3 a2
M4 a3
M5 a4
;
```

```
; Dominio dos lugares
L1 1
L2 2
L2 3
L2 4
L2 5
L3 2
L3 3
L3 4
L3 5
;
; Alteracoes definidas
A1 4 cx1  1 0 1
A2 5 cx2  1 1 0
A3 6 cx3;a1  2 2 0  1 1 -1  3 0 2
A4 6 cx3;a2  2 3 0  1 1 -1  3 0 3
A5 6 cx3;a3  2 4 0  1 1 -1  3 0 4
A6 6 cx3;a4  2 5 0  1 1 -1  3 0 5
A7 7 cx4;a1  3 2 0  2 0 2
A8 7 cx4;a2  3 3 0  2 0 3
A9 7 cx4;a3  3 4 0  2 0 4
A10 7 cx4;a4  3 5 0  2 0 5
;
; fim do arquivo
```

A.3 Rede MORTA.DAT

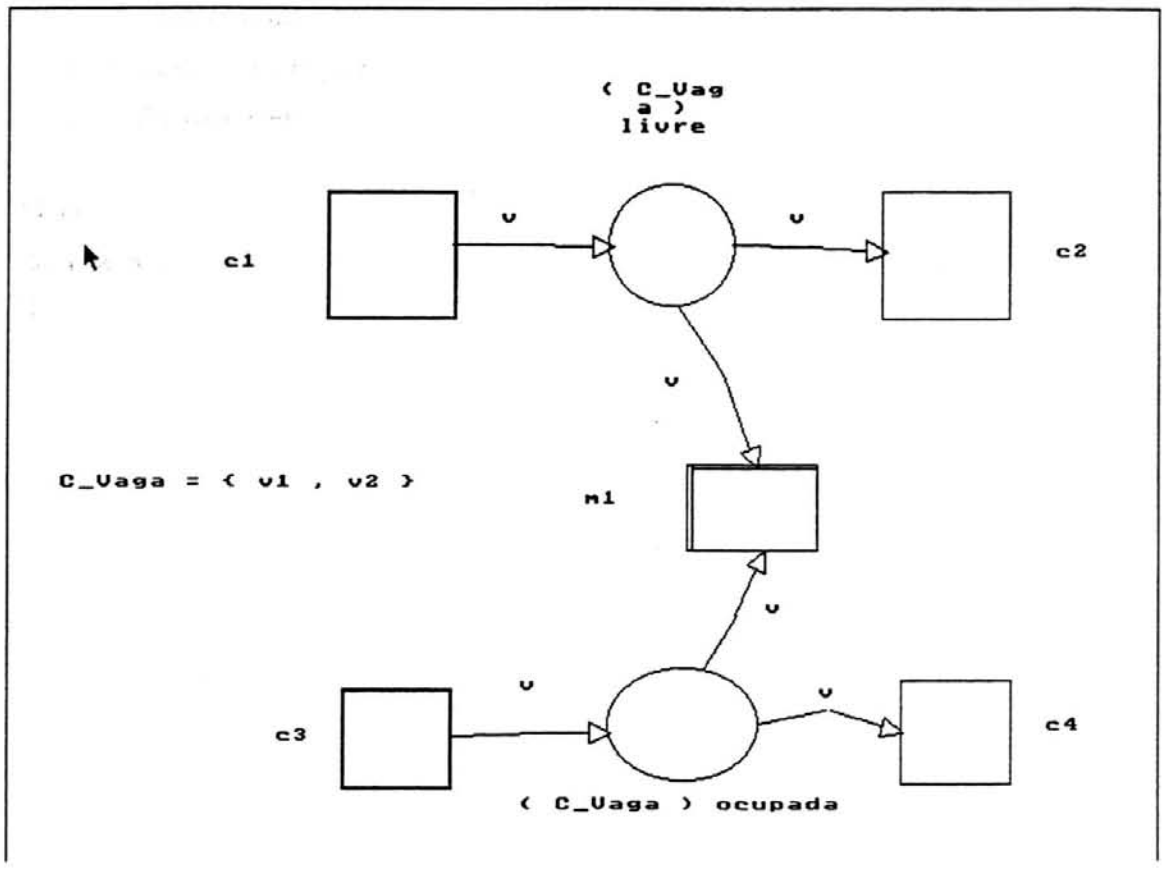


Figura A.3 Rede MORTA.DAT

```
; Arquivo gerado pelo simulador didatico de redes
; de Petri de alto nivel.
;
; Flavio Soibelmann Glock 1990/1991
;
; Nome do arquivo: morta
;
; Geometria da rede
E1 -20104 6450 -20028 6525  -20157 6398 -19975 6578
R2 -20273 6455 -20196 6532  -20290 6437 -20178 6550
R3 -19940 6455 -19864 6532  -19958 6437 -19846 6550
F4 -20057 6622 -19978 6672  -20112 6587 -19923 6707
```

```
X5 -19984 6488 -19937 6491
X6 -20159 6487 -20101 6488
X7 -20036 6567 -20014 6624
N8 -20188 6443 -20137 6501  -20224 6402 -20101 6542  v
N9 -20013 6441 -19965 6504  -20047 6397 -19931 6548  v
Z10 -20030 6487 -19983 6488
N11 -20374 6464 -20273 6533  -20445 6416 -20202 6581  c1
N12 -19869 6460 -19783 6525  -19929 6415 -19723 6571  c2
N13 -20119 6362 -20023 6452  -20186 6299 -19956 6515  ( C_Vaga )
livre
Z14 -20198 6487 -20157 6487
N15 -20459 6584 -20189 6684  -20648 6514 -20000 6754  C_Vaga = {
v1 , v2 }
Z16 -20062 6525 -20036 6567
N17 -20110 6543 -20018 6601  -20174 6502 -19954 6642  v
N18 -20164 6623 -20050 6664  -20244 6594 -19970 6693  m1
R19 -20264 6756 -20198 6817  -20310 6713 -20152 6860
R20 -19928 6751 -19863 6814  -19974 6707 -19818 6858
N21 -19866 6754 -19797 6806  -19914 6718 -19749 6842  c4
N22 -20338 6756 -20244 6817  -20404 6713 -20178 6860  c3
E23 -20104 6743 -20014 6812  -20167 6695 -19951 6860
N24 -20177 6809 -19941 6848  -20342 6782 -19776 6875  ( C_Vaga )
ocupada
N25 -19997 6741 -19944 6777  -20034 6716 -19907 6802  v
N26 -20174 6720 -20126 6790  -20208 6671 -20092 6839  v
N27 -20043 6691 -19962 6727  -20100 6666 -19905 6752  v
Z28 -20046 6743 -20027 6712
X29 -20028 6712 -20010 6672
Z30 -20143 6783 -20198 6785
X31 -20143 6783 -20102 6783
Z32 -20014 6777 -19973 6769
X33 -19970 6769 -19927 6783
;
; Universo de discurso
M1 v1
M2 v2
```

```
;
; Dominio dos lugares
L1 1
L1 2
L23 1
L23 2
;
; Alteracoes definidas
A1 2 c1;v1  1 0 1
A2 2 c1;v2  1 0 2
A3 3 c2;v1  1 1 0
A4 3 c2;v2  1 2 0
A5 4 m1;v1  1 1 0  23 1 0
A6 4 m1;v2  1 2 0  23 2 0
A7 19 c3;v1  23 0 1
A8 19 c3;v2  23 0 2
A9 20 c4;v1  23 1 0
A10 20 c4;v2  23 2 0
;
; fim do arquivo
```

A.4 Rede MULTI.DAT

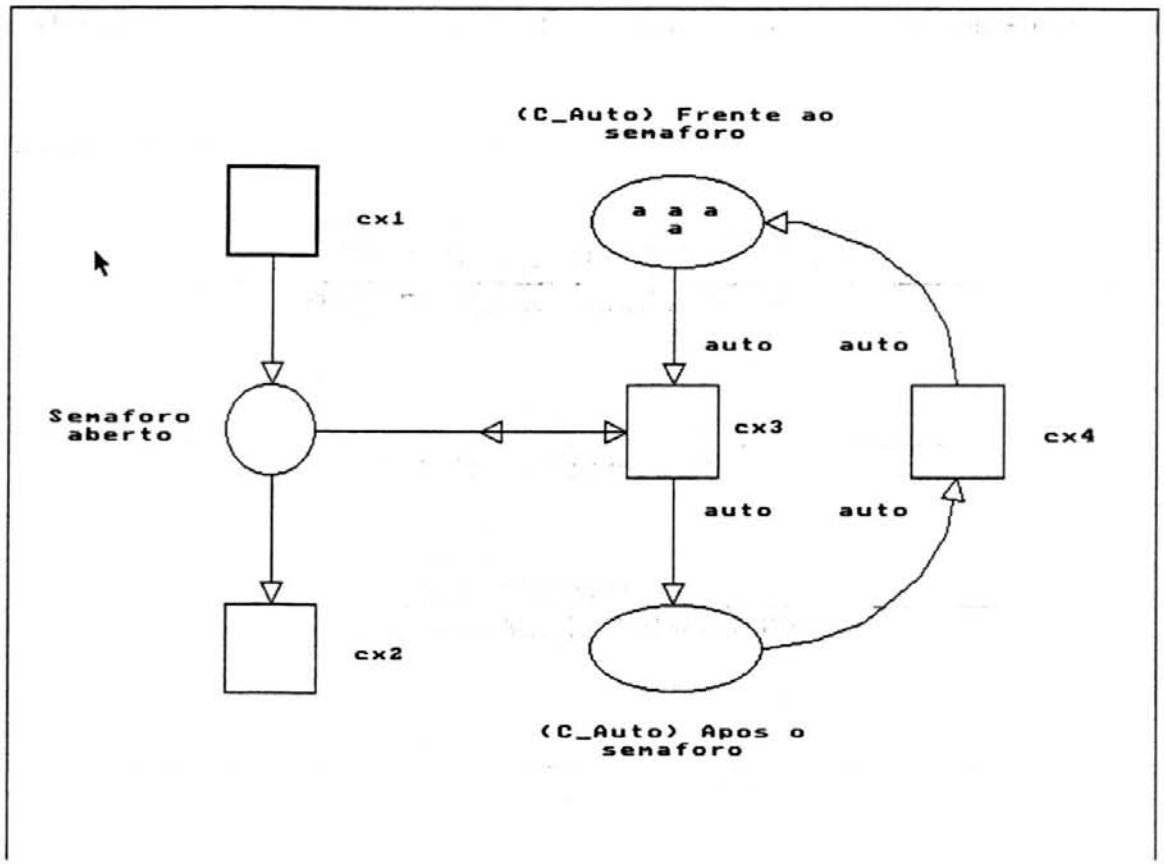


Figura A.4 Rede MULTI.DAT

; MULTI.DAT
; Titulo: Descricao do desenho de uma rede de Petri: "Modelo do Semaforo"
; Autor: Flavio Soibelmann Glock
; Data: 25 de setembro de 1990
; Modificado: 27 de abril de 1991
;
; geometria
; E = elipses
; R = retangulos
; F = "fact"
; N = notas
; X = setas

```
; Z = linhas
;
;         LUGARES (elipses)
;
;  coordenadas      area de interesse
E1  70  27  80  37    48  13  95  51
E2 110   4 129  14    99   4 140  27
E3 110  51 129  61    99  37 140  72
;
;       CONEXOES (retangulos)
;
;  coordenadas      area de interesse       anotacao
R4  70   3  80  13    70   3  95  27
R5  70  51  80  61    70  37  95  61
R6 114  27 124  37    98  14 139  51
R7 145  27 155  37   129  14 170  61
;
;        NOTAS
;
;  coordenadas      area de interesse       anotacao
N8  80   5  95  13    70   3  95  13   cx1
N9  48  26  70  38    48  26  75  37   Semaforo aberto
N10 80  53  95  61    70  51  95  61   cx2
N11 99  -7 140   4    99  -7 140  14   (C_Auto) Frente ao semaforo
N12 119  19 134  27    119  19 134  27   auto
N13 124  27 134  37    124  27 134  37   cx3
N14 119  37 134  45    119  37 134  45   auto
N15  99  61 140  72    99  51 140  72   (C_Auto) Apos o semaforo
N16 134  19 149  27    134  19 149  27   auto
N17 134 37 149  45    134  37 149  45   auto
N18 155 29 170  37    145  27 170  37   cx4
;
X19   75  13  75  27
X20   75  37  75  51
X21   80  32 114  32
X22  114  32  98  32
```

```
X23  119  14 119  27
X24  119  37 119  51
X25  133   9 129   9
X26  149  43 150  37
;
Z27  150  27 149  21
Z28  149  21 146  16
Z29  146  16 141  12
Z30  141  12 133   9
;
Z31  149  43 146  48
Z32  146  48 140  53
Z33  140  53 135  55
Z34  135  55 129  56
;
; Descricao do funcionamento
;
; conjuntos
;
; C_Auto     1 2 3 4
;
; marcas definidas para qualquer lugar (M0 = {})
;  "M"-numero-da-marca
;  Conteudo-da-marca : string
;
;
M1 a
M2 a
M3 a
M4 a
M5 ~
;
; marcas definidas para um lugar
;  "L"-Numero-do-lugar
;  lista-de-marcas
;
```

```
L1 5
L2 1
L2 2
L2 3
L2 4
L3 1
L3 2
L3 3
L3 4
;
; alteracoes definidas
;  "A"-Numero-da-Alteracao
;  Numero da conexao
;  triplas: Numero do Lugar
;           Marca-antes
;           Marca-depois
;
; conexao 1: entra no lugar 1
;
A1  4  cx1  1 0 5
;
; conexao 2: sai do lugar 1
;
A2  5  cx2  1 5 0
;
; conexao 3: sai de 2, entra em 3, testa 1
;
A3  6  cx3;a  2 1 0  3 0 1  1 5 -1
A4  6  cx3;a  2 2 0  3 0 2  1 5 -1
A5  6  cx3;a  2 3 0  3 0 3  1 5 -1
A6  6  cx3;a  2 4 0  3 0 4  1 5 -1
;
; conexao 4 sai de 3, entra em 2
;
A7  7  cx4;a  2 0 1  3 1 0
A8  7  cx4;a  2 0 2  3 2 0
```

```
A9  7  cx4;a  2 0 3  3 3 0
A10 7  cx4;a  2 0 4  3 4 0
;
; marcacao inicial
;  "I"-numero-do-lugar
;  lista-de-marcas
;
I1 0
I2 1 2 3 4
I3 0
```

BIBLIOGRAFIA

[ALB 89] ALBERT, K. et al. DESIGN/CPN: a tool package supporting the use of colored Petri nets. **Petri Net Newsletter**, Bonn, n.32, p.22-36, Apr. 1989.

[BIL 88] BILLINGTON, J. et al. PROTEAN: a high level Petri net tool for the specification and verification of communication protocols. **IEEE Transactions on Software Engineering**, v.14, n.3, Mar. 1988.

[BRA 87] BRAUER, W. **Preface.** Berlin: Springer-Verlag, 1987. p.3-5. (Lecture notes in computer science, v.254).

[CIN 86] DE CINDIO, F.; POMELLO, L. The second advanced course on Petri nets. **Petri Net Newsletter**, Bonn, n.25, Dec. 1986.

[GEN 81] GENRICH, H. J.; LAUTENBACH, K. System modelling with high level Petri nets. **Theor. Comp. Science**, n.13, p.109-136, 1981.

[GEN 87] GENRICH, H. J. **Predicate transition nets**. Berlin: Springer-Verlag, 1987. p.207-247. (Lecture notes in computer science, v.254).

[GER 87] GERSTING, J. L. Mathematical structures for computer science. New York: W. H. Freeman, 1987. p.197-200.

[GER 87a] GERSTING, J. L. **Mathematical structures for computer science**. New York: W. H. Freeman, 1987. p.202-204.

[HEU 90] HEUSER, C. A. Modelagem conceitual de sistemas: Redes de Petri. In: ESCOLA BRASILEIRO-ARGENTINA DE INFORMÁTICA, 5., 1991, Nova Friburgo. **Anais**... Nova Friburgo: EBAI, 1991. 150p.

[HIN 90] HINTZ, E. C. **Um estudo sobre o uso do DESIGN/OA no auxílio a metodologias de análise e projeto de sistemas de informação**. Porto Alegre: UFRGS - Instituto de Informática, 1990. 120p. Bacharelado em Ciências de Computação.

[JEN 84] JENSEN, K. **The design of a program package for an introdutory Petri net course**. Berlin: Springer-Verlag, 1984. p.259-266. (Lecture notes in computer science, v.188).

[JEN 87] JENSEN, K. **Coloured Petri nets**. Berlin: Springer-Verlag, 1987. p.248-299. (Lecture notes in computer science, v.254).

[MAZ 90] MAZIERO, C. A. **Um ambiente para a análise e simulação de sistemas modelados por redes de Petri**. Florianópolis: UFSC, 1990.

[PET 76] PETRI, C. A. **Interpretations of net theory**. 2. ed. Bonn: Gesselschaft für Math. und Datenverarbeitung mbh Bonn, 1976. (Technical Report ISF 75-07).

Informática
UFRGS

"Um simulador de redes de Petri de alto nível para uso didático".

Dissertação apresentada aos Srs.:

Prof. Dr. Carlos Alberto Heuser

Prof. Dr. Cláudio Walter

Prof. Dr. Jean-Marie Farines (UFSC)

Prof. Dr. Raul Fernando Weber

Vista e permitida a impressão.
Porto Alegre, 14/ 11/ 97.

Prof. Dr. Carlos Alberto Heuser,
Orientador.

Prof. Dr. Ricardo A. da L. Reis,
Coordenador do Curso de Pós-Graduação
em Ciência da Computação.