UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA
DEPARTAMENTO DE TELEMÁTICA

UM MÉTODO PARA A VALIDAÇÃO DE PROTOCOLO DE COMUNICAÇÃO ESPECIFICADO EM SDL UTILIZANDO REDE DE PETRI

Autor : Roberto Toshiyuki Tamura

Orientador : Prof. Dr. Shusaburo Motoyama

Parecer de Tese
Este exemplar corresponde à redação final da tese defendida por Roberto Toshiyuki Tamura e aprovada pela comissão julgadora em 02 setembro de 1988
Shusa Motoyama

Tese apresentada à Faculdade de Engenharia Elétrica como parte dos requisitos exigidos para a obtenção do Título de Mestre em Engenharia Elétrica.

Setembro 1988

Aos

meus pais

AGRADECIMENTOS

Ao Motoyama pelo assunto da tese, constante trabalho de orientação, estímulo e paciência.

Ao Borelli e Mauro pelas inúmeras contribuições referentes a este trabalho.

À Neli e Huda pela dedicação e ajuda relevante.

Aos amigos da UNICAMP, em especial os pertencentes ao Departamento Telemática pelos incentivos e agradável convivência.

I N D I C E

CAPÍTULO 1

INTRODUÇÃO

Uma rede de comunicação envolve a troca de informações entre vários entidades. Chama-se de entidade a qualquer componente de um sistema ("software" ou "hardware") capaz de produzir e/ou consumir informações. Para que esta troca de informações possa ser feita de forma ordeira, um conjunto de regras se faz necessário. A este conjunto de regras denomina-se protocolo.

Protocolos são necessários para regulamentar uma série de aspectos relacionados à troca de informações. Inicialmente, faz-se necessário convencionar qual a unidade de informações que vai ser trocada entre as entidades participantes. No entanto, para descrever a interação entre dois processos executados em computadores diferentes, é mais conveniente olhar para a unidade de transferência como sendo caracter, ou mensagem, ou arquivos, etc. O protocolo é então responsável por introduzir o nível de abstração necessário para descrever a interação entre as entidades, em cada caso.

Um segundo aspecto definido pelo protocolo concerne à criação de convenções, tais como a definição do código de representação das unidades sendo trocadas, os formatos usados, quais as velocidades e que controle podem ser usados para controlar a transferência.

O projeto de protocolo de uma rede de comunicação que

permita a seus usuários se comunicarem eficientemente é uma tarefa muito complexa, pois deve considerar todos os eventos possíveis de acontecer durante a comunicação e conhecer todos os efeitos.

Na descrição do protocolo pode-se usar uma linguagem natural, por exemplo a portuguesa. No entanto, embora dêem a impressão de fácil compreensão, pode resultar em descrições informais as quais geralmente têm os atributos de serem incompletas e longas, não consistente, de compreensão dúbia ou de interpretação variada. É óbvio que um protocolo não pode ser descrito informalmente sob o risco de não cobrir todas as possíveis situações ou estados da comunicação a ser controlada, de funcionar indevidamente sob certas condições e de ser implementado em sistemas distintos por equipes diferentes com respectivas interpretações.

É indispensável que a especificação ou descrição de um protocolo seja concisa e precisa, totalmente ausente de ambiguidade. Para este objetivo, existem na literatura vários métodos propostos para especificação formal de protocolos.

De uma forma geral, pode-se classificar os métodos propostos em três categorias : modelos de transição, linguagens de programação e modelos híbridos ou mistos.

Os modelos de transição se baseiam no reconhecimento de que as entidades participantes de um protocolo têm seus comportamentos regidos por reações a determinados eventos, tais como a chegada de uma mensagem, uma temporização, etc. Este modelo de especificação pode envolver apenas uma máquina de estados finitos para todo o sistema, ou pode ser construído por

pares de máquina, onde as transição em cada uma são sincronizadas.

Os métodos de linguagens de programação são baseados no fato de que protocolos não deixam de ser um tipo de algorítmos concorrentes, e portanto, podem ser especificados através de programação. Várias linguagens já foram usadas com este propósito, tais como PASCAL (estendido), PROLOG, etc.

Os métodos híbridos procuram combinar os métodos anteriores. Assim, o comportamento da entidade é descrito por uma pequena máquina de estado finito, aumentada por especifições em alguma linguagem de programação onde os efeitos dos eventos sobre as variáveis locais de cada entidade são descritos.

A especificação formal de um protocolo não garante, por si só, que o protocolo não venha a funcionar de maneira indesejada ou indevida. Isto porque erros podem ter sido introduzidos na especificação ou a especificação pode ter sido incompleta no sentido de não cobrir todas as possíveis situações de operações do protocolo. É necessário, então, que a especificação seja validada para garantir que ela seja livre de defeitos ou de propriedades anômalas.

Dada a complexidade de grande número de protocolos usados na prática, têm-se buscados métodos formais de validação que permitam ao projetista do protocolo certificar-se de sua correção. A maior parte dos trabalhos que têm sido feitos se concentram na verificação de propriedades do protocolo. Dentre estas propriedades, existem algumas que são comuns a todos os protocolos, tais como ausência de impasses, e de progresso

efetivo (presença de "loops"). A ausência de impasses significa que, a cada instante, existe alguma ação ou evento que pode ocorrer, a menos que o sistema se encontre em um estágio final aceitável, onde o objetivo do protocolo foi atingido.

A ausência de impasse é essencial para o funcionnamento correto do protocolo. No entanto, um sistema isento de impasse não quer dizer, necessariamente, que está funcionalmente correto. Para que isto aconteça, é necessário que o sistema, funcionando, atinja seu objetivo eventualmente. Em sistemas onde existem paralelismo, pode acontecer que este entre num ciclo improdutivo, de forma que, embora a cada instante exista algum processamento sendo feito, o sistema não atinge o seu objetivo final.

Os protocolos especificados em modelos de transição podem ser validados através da geração sistemática de todos os estados possíveis do sistema. A maior dificuldade encontrada por este métodos é a chamada "explosão de estados" que é causada pelo grau de número de combinação de transições que podem ocorrer.

Os protocolos especificados em linguagens de programação podem ser validados usando as técnicas de verificação de programas. Estas técnicas podem, potencialmente, lidar com todos os tipos de propriedades que se desejam provar, mas a sua automatização é bastante difícil.

Os métodos híbridos de validação usam a combinação das duas técnicas descritas acima; o uso de variáveis simbólicas, por exemplo, permite a redução do número total de estados possíveis do sistema. Estas técnicas, também, são difíceis de automatizar completamente.

Neste trabalho, o método formal de validação de protocolo é

baseado em modelo de transição. O método utiliza uma linguagem de projeto chamada SDL (Specification and Description Language) [1], para a especificação formal do protocolo. E para a verificação das propriedades do protocolo é utilizada a rede de Petri [2,3,13].

No capítulo 2 são apresentadas as principais definições e propriedades da rede de Petri e da linguagem SDL. É discutida também uma técnica de redução de rede de Petri [3,4,5,13] para minimizar o problema de "explosão de estados".

No capítulo 3 são apresentadas as regras para realizar a conversão de protocolos especificados em linguagem SDL para a sua representação em Rede de Petri equivalente [6,7]. São discutidas as implementações do conversor automático que realiza a conversão de protocolo especificado em SDL para Rede de Petri, e o analisador automático de rede de Petri. É apresentada a estrutura do programa e são discutidos em detalhes os algorítmos implementados.

No capítulo 4 é apresentado um exemplo de verificação das propriedades de um protocolo utilizando a ferramenta acima citada.

Finalmente, no capítulo 5 são apresentadas as principais conclusões do trabalho.

CAPÍTULO 2

CONCEITOS BÁSICOS DE REDE DE PETRI E REGRAS E PROCEDIMENTOS DA LINGUAGEM SDL

2.1 - Introdução

Neste capítulo, são apresentados inicialmente as regras e procedimentos básicos da linguagem de especificação e descrição de sistemas - LEDS (ou SDL - Specification and Description Language) [1], utilizada na representação gráfica de processos em geral. A SDL utiliza uma representação em forma de gráfico, e é muito útil, pois pode representar processos concorrentes.

Em seguida, são apresentadas as principais definições de rede de Petri e as principais propriedades gerais [2,3,9]. É discutida, também uma técnica de redução de lugares de rede de Petri [3,4,5,13], para minimizar o problema de aumento exagerado de estados que ocorrem com frequência em redes de Petri.

2.2 - SDL : Regras e Procedimentos

A linguagem SDL foi apresentada inicialmente pelo CCITT em 1976, nas recomendações Z.101 a Z.103 para a descrição e especificação do sistema de comutação temporal controlado por programa armazenado (CPA-T). Em 1980 foi redefinida nas recomendações Z.101 a Z.104 incorporando novas estensões.

A linguagem SDL descreve o comportamento do sistema na forma estímulo/resposta (entrada/saída). Portanto, a representação de um sistema em SDL é baseada no conceito de máquina de estado

finito, mas incorporando regras próprias.

Como exemplo dos possíveis áreas de aplicação da SDL temos o protocolo de comunicação, sinalização de chamadas telefônica, sistemas de controle, etc. Neste trabalho os conceitos da SDL serão aplicados visando a sua utilização em protocolo de comunicação (ou apenas protocolo).

A SDL foi desenvolvida com vários recursos objetivando facilitar o usuário (autor) a especificar e descrever os sistemas (protocolos) de um modo claro e objetivo, e para facilitar a compreensão de leitura.

Um dos recursos é a divisão dos sistemas (protocolos) em forma de blocos, denominado de Blocos Funcionais (BF). Desta forma, a descrição e especificação do protocolo podem ser feitas em várias partes separadamente, com a visualização e detalhamento melhorados.

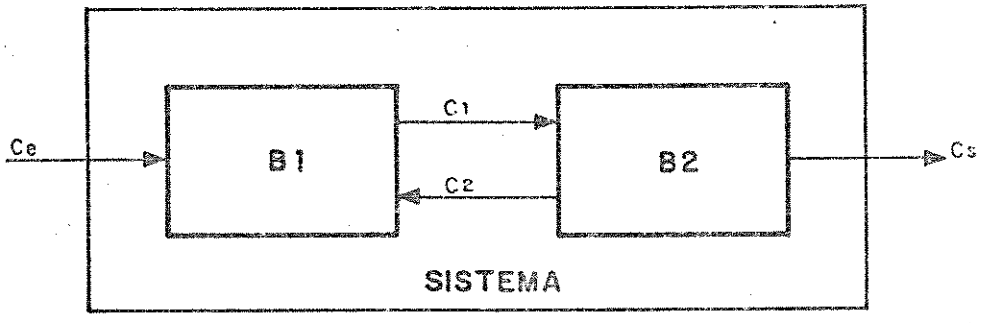


Fig. 2.1 - Exemplo de um sistema dividido em blocos.

Na Fig. 2.1 é mostrado um exemplo de um protocolo de comunicação (ou sistema) dividido em dois Blocos Funcionais representado, por B1 e B2. Estes Blocos Funcionais podem trocar

os sinais entre si representado por C1 e C2, e com o meio externo representado por Ce e Cs.

O protocolo inicial que será descrito e especificado em SDL, pertence a um nível superior, e os BF's que o compõe pertencem ao nível imediatamente inferior. Pelo mesmo motivo que se levou a dividir o protocolo em BF's, cada um destes BF's pode ser particionado em outros sub-blocos que pertencem a um nível inferior a dos BF's. Analogamente, cada um destes sub-blocos pode ser particionado em outros conjuntos de sub-blocos dentro de um nível ainda inferior.

Um BF pode ser particionado em tantos nível quanto necessários, conforme o detalhamento exigido para descrever claramente o protocolo (sistema).

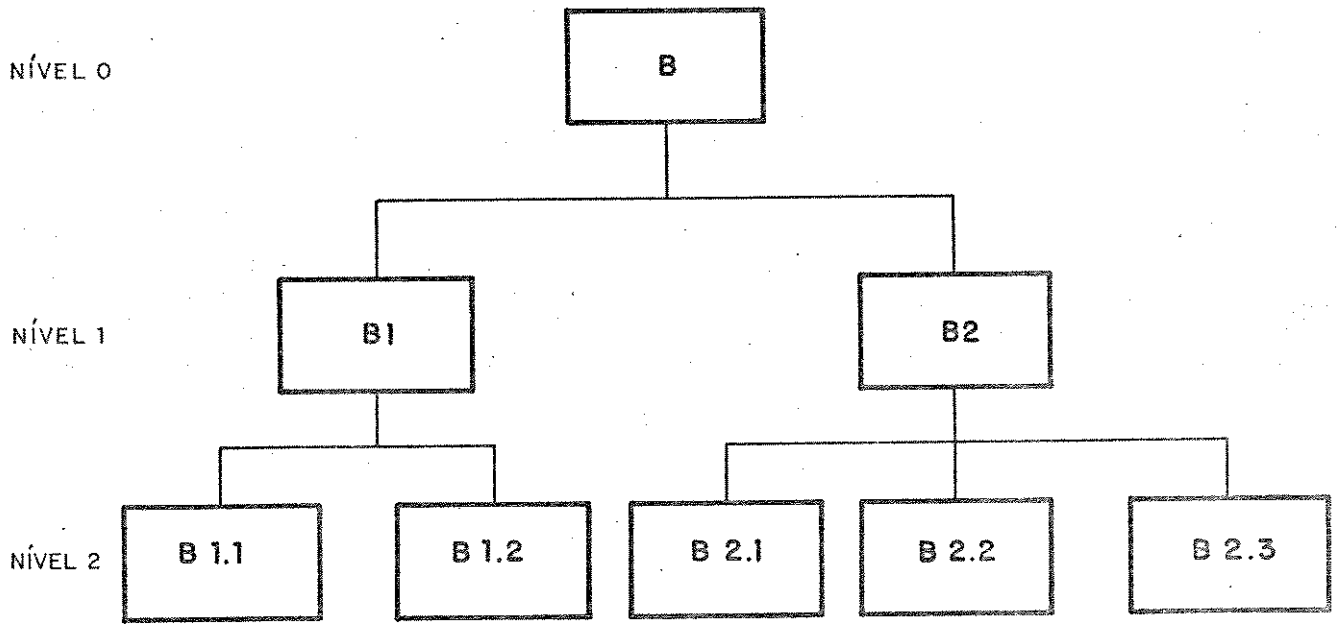


Fig. 2.2 - Exemplo de particionamento de sistema.

Na Fig. 2.2 é apresentado um exemplo da diagrama de particionamento onde o bloco B representa o nível superior

conforme indicado na figura como nível 0. Este bloco é particionado em sub-blocos representados pelos blocos B1 e B2 que pertencem a um nível imediatamente inferior indicado por nível 1.

Os blocos que pertencem a nível 1 foram particionado novamente em B1.1, B1.2, B2.1, B2.2 e B2.3 , formando o nível 2. Os sub-blocos B1.1 e B1.2 são originários do bloco B1 enquanto que o conjunto de sub-blocos formados por B2.1, B2.2 e B2.3 são originários do bloco B2.

Vários critérios podem ser utilizados para o particionamento. O particionamento pode ser feito se satisfizer pelo menos um dos seguintes casos :

a) Criar uma correspondencia com a divisão do software e/ou hardware implementado.

b) Reduzir a interação entre os blocos funcionais.

c) Definir o BF a um tamanho relativamente pequeno para facilitar a descrição.

d) Aproveitar uma representação já existente.

O comportamento dinâmico de um BF, independente do seu nível, pode ser modelado através de uma ou mais máquina de estado finito estendido denominado de processos. Estes processos são representados em termos de Sinal, de Estado, de Transição, de Entrada, de Adiamento (Save), de Decisão, de Tarefa e de Saídas, onde:

Sinal - É uma representação do fluxo de dados ou informações trocadas entre os processos. Quando os dados trocados são entre os processos que pertencem a BF's diferentes são sinais externos,

e quando são trocados entre processos que pertencem ao mesmo BF são sinais internos.

ENTRADA - É um sinal que chega ao processo e que deve ser reconhecido e identificado pelo mesmo. De acordo com a definição do sinal, a entrada pode ser interna ou externa.

ESTADO - É uma condição do processo em que a ação está suspensa a espera de uma entrada.

ADIAMENTO - Indica um armazenamento temporário de um sinal recebido.

TRANSIÇÃO - É uma sequência de ações tomada em resposta a um sinal de entrada. Após a transição o processo muda de um estado para outro.

SAÍDA - É uma ação durante a transição com a função de enviar um sinal para outro processo. De acordo com a definição de sinal, a saída pode ser interna ou externa.

TAREFA - Indica um conjunto de ações durante a transição que não tem efeito direto no comportamento do processo.

DECISÃO - Indica uma ação durante a transição onde se realiza uma pergunta. A sequência de ações restantes depende da resposta a esta pergunta.

Para representação gráfica do processo em termos de Estado, Entrada e Transição, utiliza-se a seguinte simbologia:

SÍMBOLO DE ESTADO ----------

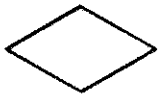
SÍMBOLO DE ENTRADA INTERNA ------

SÍMBOLO DE ENTRADA EXTERNA ------

SÍMBOLO DE ADIAMENTO ----------

SÍMBOLO DE TAREFA ----------

SÍMBOLO DE DECISÃO ----------



SÍMBOLO SAÍDA INTERNA ----------

SÍMBOLO SAÍDA EXTERNA ----------

Para representar o processo utilizando estes símbolos, deve-se obedecer as seguintes regras :

a) Um símbolo de estado deve ser seguido apenas pelo símbolo de entrada ou símbolo de adiamento.

b) Os símbolos de entrada e de adiamento devem ser precedidos pelo símbolo de estado.

c) O símbolo de entrada precede apenas um único símbolo que pode ser de qualquer tipo exceto símbolos de entrada e de adiamento.

d) O símbolo de adiamento não deve preceder nenhum símbolo.

e) Um símbolo de tarefa ou um símbolo de saída precede apenas um único símbolo, que não pode ser símbolo de adiamento e de

entrada.

f) O símbolo de decisão pode preceder quaisquer símbolos que não sejam os símbolo de adiamento e de entrada.

g) Todos os símbolos são interligados com o seu símbolo precedente através da linha de fluxo.

h) A convergencia da linha de fluxo e/ou o conector é usado quando um símbolo tem vários símbolos precedentes.

i) Quando um símbolo precede dois ou mais símbolos utiliza-se a divergência da linha de fluxo.

j) Entre um símbolo de saída de um processo e um símbolo de entrada de outro processo que sejam correspondentes, pode ser utilizada a linha de sinal para indicar a sua associação.

k) Qualquer símbolo ou linha de fluxo pode conter um comentário.

l) Todos os símbolos devem ter títulos (nomes) para identificar as suas funções no processo.

Em SDL, um processo é representado por uma rede de estados e transições. O estado é um ponto no processo em que não está executando nenhum tipo de ação mas fica monitorando a chegada de sinal do outro processo. Se o sinal que chegar coincidir com o nome do símbolo de entrada que sucede o estado, o processo sai do estado de monitoração da chegada de sinal de entrada e começa a executar a transição após consumir este sinal.

Dois símbolos de estado em um mesmo diagrama SDL deve ter nomes diferentes atribuidos a eles se ocorrerem pelo menos um dos seguintes casos :

a) Os nomes e/ou número (quantidade) de símbolos de entrada

e de adiamento que os sucedem são diferentes.

b) As sequências de ações para serem executadas nas transições em resposta a uma entrada são diferentes.

c) Os estados atingidos após a execuções das transições são diferentes.

Exemplo 2.1 - APLICAÇÃO EM UM PROTOCOLO DE SUSPENSÃO E DE RELIGAÇÃO DE CHAMADAS (TELEFONICA OU DE DADOS) NA RDSI (REDE DIGITAL DE SERVIÇOS INTEGRADOS).

A seguir, é descrito em SDL um protocolo utilizado em RDSI (Recomendação provisória do sistema Telebrás) para suspensão e religação de chamada telefônica ou de dados.

O protocolo proposto representa uma fase em que o assinante está conectado à rede, mas o usuário deseja uma suspensão temporária de chamada e posteriormente solicita a religação da chamada. Este protocolo pode ser dividido em dois blocos funcionais, cada um com um processo (Fig. 2.3).

- Processo 1 representa os estados de solicitação de suspensão e de religação de chamada do lado de usuário.

- Processo 2 representa os estados envolvidos no lado da rede.

Sejam os seguintes parâmetros envolvidos durante a suspensão e religação.

a) Estados do lado usuário.

. Ativo - Usuário

. Pedido de Suspensão

. Suspensão - Usuário

. Pedido de Religação

b) Estados do lado rede.

. Ativo – Rede

. Pedido de Suspensão

. Suspensão – Rede

. Pedido de Religação

c) Sinais de saída do lado usuário (Sinais de entrada no lado rede)

. Solicitação de Suspensão

. Solicitação de Religação

d) Sinais de saída do lado rede (Sinais de entrada no lado usuário).

. Reconhecimento de Suspensão

. Rejeição de Suspensão

. Reconhecimento de Religação

. Rejeição de Religação

O usuário no estado Ativo-Usuário faz a solicitação de suspensão enviando o sinal de Solicitação de Suspensão a rede.

A rede que está no estado Ativo-Rede, ao receber o sinal de Solicitação de Suspensão pode responder de seguinte modo :

a) envia o sinal de Reconhecimento de Suspensão e fica no estado Suspensão-Rede, ou

b) envia o sinal de Rejeição de Suspensão e volta para o estado Ativo-Rede.

O usuário ao receber o sinal de Rejeição de Suspensão volta para o estado Ativo-Usuário. Por outro lado, se o sinal recebido for Reconhecimento de Suspensão, o usuário fica no estado Suspensão-Usuário.

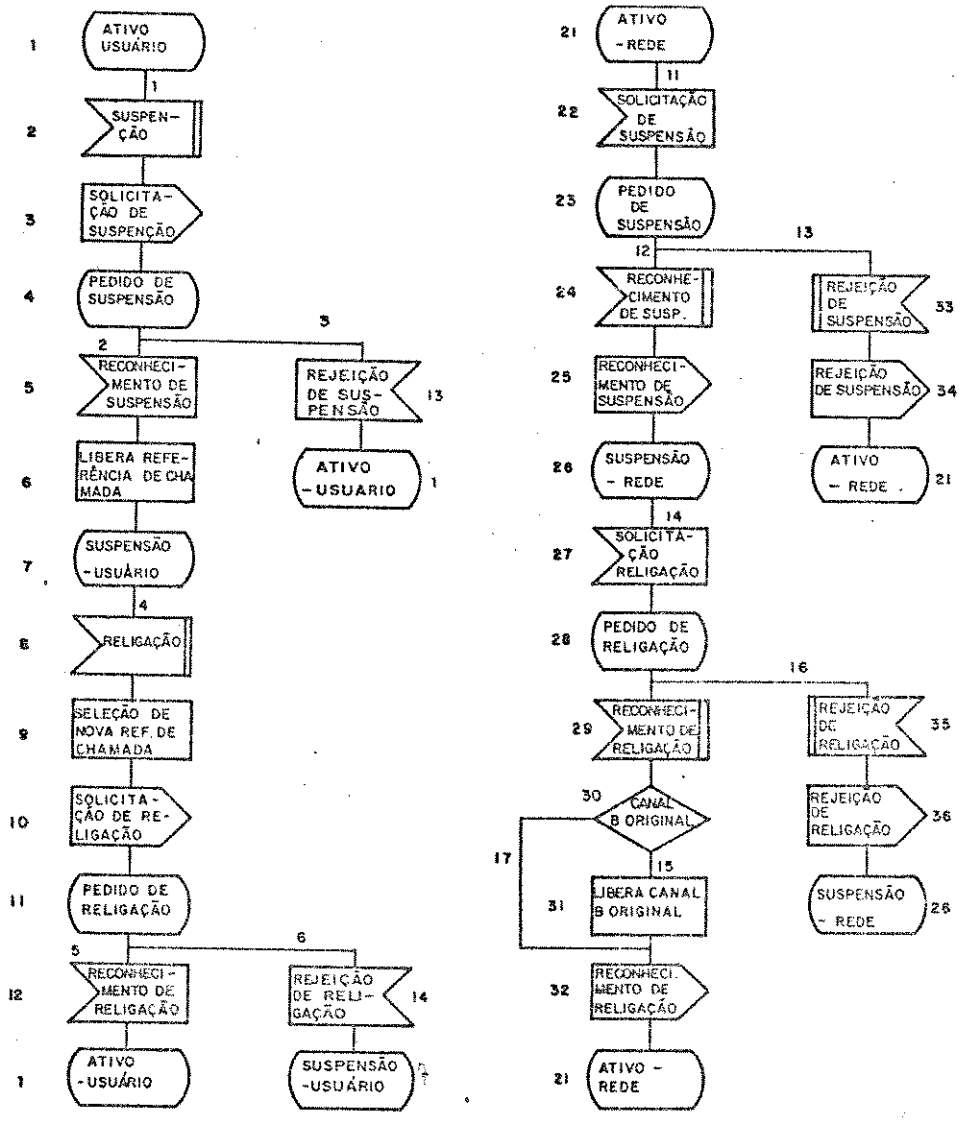


Fig. 2.3 - Suspensão e Religação de chamada : Lado Usuário e Lado Rede.

O usuário no estado Suspensão-Usuário envia o sinal de Solicitação de Religação à rede. A rede que está no estado Suspensão-Rede recebe o sinal e pode responder com Reconhecimento de Religação e voltar para o estado Ativo-Rede ou responder com Rejeição de Religação e continuar no estado Suspensão-Rede. O usuário recebe o sinal de Reconhecimento de Religação e volta para o estado Ativo-Usuário; e se o sinal for Rejeição de Religação o usuário continua no estado Suspensão-Usuário.

2.3 - Rede de Petri : Conceitos Básicos

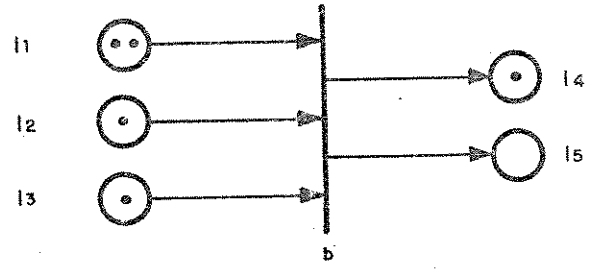
2.3.1 - Representação Gráfica e Matricial

Na representação gráfica, uma rede de Petri consiste de lugares (places) representada por circulos e barras de transições (ou simplesmente barra) que são interligados por arcos orientados. Um arco orientado inteliga um lugar a uma barra de transição ou vice-versa.

Os lugares dos quais saem arcos incidentes à uma barra são chamados lugares de entrada daquela barra. Os lugares de saída de uma barra de transição são similarmente definidos como sendo àqueles lugares que são conectados à barra por arcos com origem na barra e término no lugar.

Um lugar pode estar vazio ou ocupado por uma ou mais senhas (ou fichas. Em Inglês :" tokens").

Por exemplo, na Fig. 2.4 o lugar I1 está ocupado por duas senhas, os lugares I2, I3 e I4 por apenas uma, enquanto que o lugar I5 se encontra vazio.



b : barra de transição
I1, I2, I3 : lugares de entrada
I4, I5 : lugares de saída

Fig. 2.4 - Exemplo de lugares de entrada e de saída em uma Rede de Petri.

Note que os lugares são chamados de entrada ou de saída sempre em função de uma dada barra. Portanto, um mesmo lugar é um lugar de entrada em relação à uma certa barra bi e pode ser de saída em relação à barra bj.

No caso de i = j, o mesmo lugar é o lugar de entrada e de saída, referente à mesma barra. Neste caso o lugar é chamado um lugar realimentado.

As barras de transições podem disparar, fazendo com que as senhas fluam de lugares à lugares da rede, ou seja, alterando sempre o estado (marcação) da rede.

Cada arco, que une um lugar à barra ou vice-versa, é associado à um número inteiro, chamado peso de transição.

Quando um arco une um lugar à barra, temos um peso de transição de entrada, (ou pe). O peso pe representa o número de senhas que o lugar de entrada perde caso houver o disparo da barra mencionada.

Da mesma forma, quando o arco une uma barra à um lugar de saída temos um peso de transição de saida (ou ps). O peso ps representa o número de senhas que o lugar de saída ganha, caso houver o disparo da referida barra. Quando o peso não é especificado no arco, subtende-se que o peso é 1.

Uma barra de transição obedece seguintes regras de disparo :

a) Uma barra é habilitada ou disparável se cada um dos lugares de entrada contém um número de senhas maior ou igual ao peso de transição de entrada do arco que o une à barra.

b) O disparo de uma barra habilitada consiste em remover pe senhas de cada um dos lugares de entrada e adicionar ps senhas à cada um dos lugares de saída.

Observe que na Fig. 2.5a a barra b está habilitada. Com o disparo da barra b a rede deve passar para uma nova marcação (Fig. 2.5b).

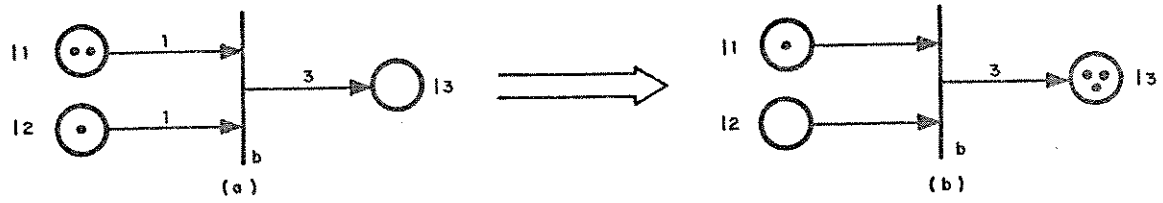


Fig 2.5 - Exemplo de barra habilitada.

Desta forma, a rede passou de uma marcação Mi para uma nova marcação Mj, através do disparo da barra b. Diz-se também, que a barra b é habilitada em Mi.

Notação: $Mi \xrightarrow{b} Mj; \quad i <> j$

Uma marcação qualquer pode ser denotada como sendo um vetor, de dimensão igual ao número total de lugares da rede, com cada componente representando o número de senhas que possui cada um dos lugares.

No exemplo da Fig. 2.5 há 3 lugares (I1, I2, I3).

Portanto, temos Mi = (2,1;0) e Mj = (1,0,3).

Após a conceituação inicial da rede de Petri de uma forma gráfica, passaremos agora à formalização matemática com vistas ao tratamento em computador. Para isso são utilizadas as seguintes definições:

Definição 1 : Uma rede de Petri pode ser representada por quatro variáveis (L, T, Alfa, Beta), onde:

L : é o conjunto de lugares com NL elementos.

T : é o conjunto de barras de transições com NT elementos.

Alfa : L x T -> N; é uma função de incidencia direta.

Beta : L x T -> N; é uma função de incidência reversa.

N : conjuto de inteiros positivos.

As funções de incidência mencionadas podem ser representadas de forma matricial, o que facilita bastante a manipulação da rede em computador.

Desta forma, teremos as seguintes matrizes de incidência:

A : matriz de incidência direta.

B : matriz de incidência reversa.

Definição 2 : Matriz de Incidência Direta: Matriz A

É formada por NT linhas e NL colunas. Ela dá uma indicação da estrutura da rede quanto às informações dos lugares de entrada que incidem sobre cada barra específica e seu respectivo peso da entrada.

O elemento da linha i e coluna j (i=1,2,...,NT; j=1,2,...,NL) denotado por Aij contém as seguintes informações:

a) se Aij = 0, o lugar lj não é lugar de entrada da barra bi.

b) se Aij > 0, o lugar lj é um lugar de entrada de bi com peso pe = Aij.

Definição 3 : Matriz de Incidência Reversa: matriz B

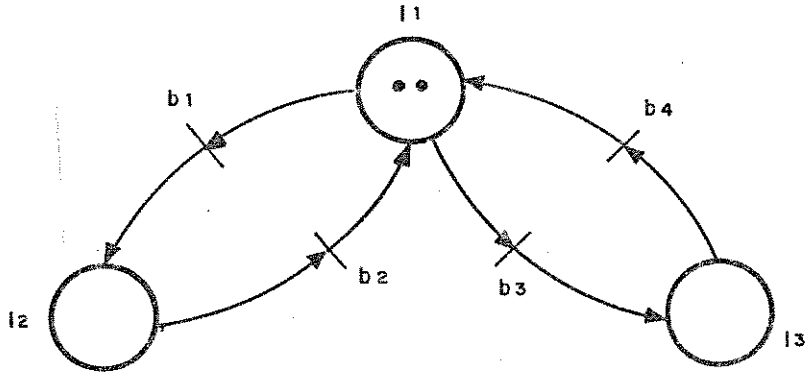
É uma matriz de dimensão idêntica à matriz A, isto é, com NT linhas e NL colunas, com a única diferença de que esta fornece uma indicação da estrutura da rede quanto a informações dos

lugares de saída que são atingidos por cada barra específica e seu respectivo peso de saída.

O elementos da linha i e coluna j (i=1,2,...,NT; j=1,2,...,NL) denotado por Bij, contém as seguintes informações:

a) se Bij = 0, o lugar não é lugar de saída da barra bi.

b) se Bij > 0, o lugar lj é lugar de saída de bi, com ps = Bij.



$$A = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{vmatrix} \qquad B = \begin{vmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{vmatrix}$$

Fig. 2.6 – Exemplo de uma rede de Petri e respectiva matriz de incidência direta e matriz de incidência reversa.

Definição 4 : Marcação de uma Rede de Petri

Dá a informação do estado da rede, ou seja, indica o número de senhas em cada lugar. Ela pode ser de 2 tipos : inicial ou decorrente como definidas a seguir.

a) Marcação inicial = M0.

É a marcação a partir da qual a rede será analisada. Em outras palavras ela corresponde ao estado inicial do sistema.

b) Marcação Decorrente.

É uma marcação qualquer da rede acessível após um ou mais disparos, partindo de uma dada marcação inicial M0.

c) Conjunto de Marcações Decorrentes de M0.

É o conjunto de todas as marcações (estados) pelas quais o sistema pode passar, sempre a partir de uma marcação inicial M0. (Notação : {D(M0)})

d) Marcação Superior

Uma marcação M' pertencente a {D(M0)} é superior à uma marcação M pertentente a {D(M0)} se e somente se

qualquer l pertencente a L, M'(l) >= M(l) e

existe lj pertencente a L, onde M'(lj) > M(lj);

onde L é o conjunto de lugares que pertencem à rede.

Notação : M' > M.

Definição 5 : Sequência de Disparos.

É a sequência de barras disparáveis, que leva a rede de uma marcação Mj à Mk, podendo inclusive j ser igual a k. A sequência pode ser representado por :

$$Mj \xrightarrow{bi} Mj+1 \xrightarrow{bj} Mj+2 \cdots\cdots \xrightarrow{bn} Mk$$

bi, bj...bn pertencem a T; onde T é o conjunto das barras que pertencem a rede;

ou resumidamente por

$$Mj \xrightarrow{Tal} Mk \qquad \text{onde } Tal = \{bi, bj \ldots bn\}$$

é a sequência de disparos das barras de transições.

Definição 6 : Habilitação de uma barra de transição

Uma barra bi de uma rede de Petri é habilitada à disparar, para uma dada marcação, se somente se,

qualquer ij pertencente a L, $M(ij) >= Aij$;

$i = 1, 2...NT$ e $j = 1, 2...NL$

Definição 7 : Ocorrência do disparo de uma barra.

O disparo de uma barra bi habilitada à disparar é definido pela transformação da marcação M em uma nova marcação M', tal que

$M'(ij) = M(ij) + Bi,j - Ai,j$;

$j = 1,....,NL$ e $i = 1,....,NT$

No exemplo da Fig. 2.6, para M = (2,0,0), o disparo de b1 resultará em :

$M'(i1) = M(i1) + B1,1 - A1,1 = 2 + 0 - 1 = 1$

$M'(i2) = M(i2) + B1,2 - A1,2 = 0 + 1 - 0 = 1$

$M'(i3) = M(i3) + B1,3 - A1,3 = 0 + 0 - 0 = 0$

Logo, o disparo de b1 é definido pela transformação

$M \xrightarrow{b1} M'$, onde M = (2,0,0) e M' = (1,1,0)

Definição 8 : Tabela de marcação

A tabela de marcação é uma tabela contendo todas as marcações decorrentes obtidas à partir da marcação inicial M0.

Estas marcações obtidas aplicando a definição 7, podem ser colocadas na forma matricial. A Fig. 2.7 mostra uma tabela de marcação obtida da rede de Petri da Fig. 2.6 com marcação inicial M0 = (2,0,0).

Marcação\Lugar	I1	I2	I3
M0	2	0	0
M1	1	1	0
M2	1	0	1
M3	0	2	0
M4	0	1	1
M5	0	0	2

Fig 2.7 - Tabela de marcações da rede de Petri da Fig. 2.6 com M0 = (2,0,0).

Definição 9 : Máquina de Senha (token machine) de uma rede de Petri

É um gráfico que mostra todas as marcações (ou estados), acessíveis numa rede de Petri, a partir de uma marcação inicial M0. O gráfico mostra as marcações representadas por círculos e arcos orientados interligando as marcações. Cada arco é rotulado pela barra cujo disparo leva uma marcação à outra.

Um exemplo de máquina de senha representada gràficamente é mostrado na Fig. 2.8.

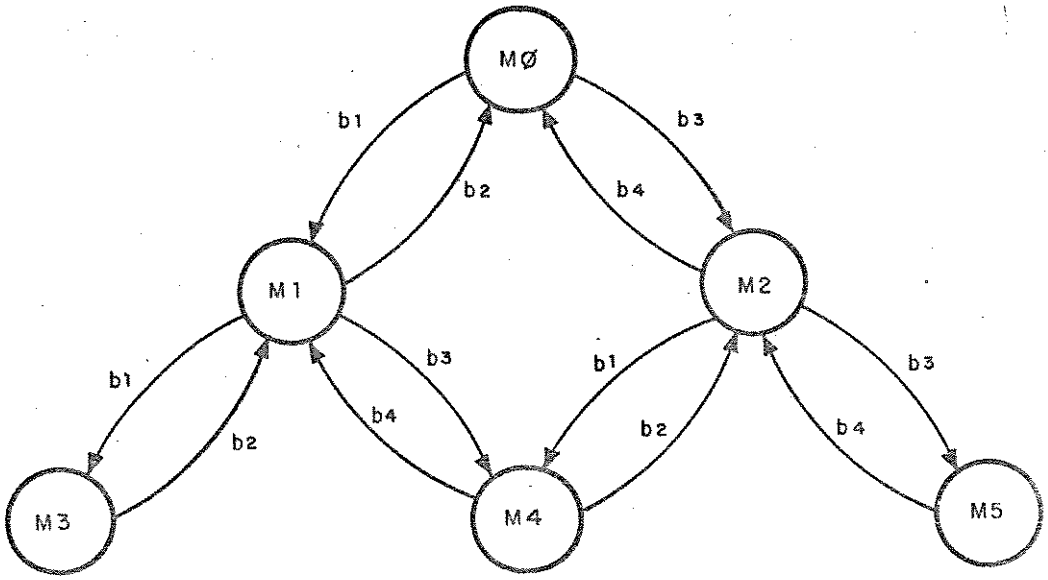


Fig. 2.8 - Máquina de Senha de RP da Fig. 2.6 com M0 = (2,0,0).

Por outro lado, para trabalhar com o computador torna-se melhor representar a máquina de senha na forma matricial com número de linhas igual a quantidade de marcações decorrentes mais um (marcação inicial) e com NT colunas. A linha define o estado que a rede está no momento e a coluna representa a barra disparada. No cruzamento destas indica a próxima marcação atingida.

A Fig. 2.9 mostra a máquina de senha da Fig. 2.8 representada na forma matricial.

Marcação\Barra	b1	b2	b3	b4
M0	1	-	2	-
M1	3	0	4	-
M2	4	-	5	0
M3	-	1	-	-
M4	-	2	-	1
M5	-	-	-	2

Fig. 2.9 - Máquina de Senha da Fig. 2.8 representada na forma de matriz.

2.3.2 - Propriedades Gerais da Rede de Petri Clássica

As principais propriedades da rede de Petri são:

a) Limitabilidade

Uma rede de Petri é definida como limitada a um valor n (n inteiro) se para qualquer marcação M decorrente da marcação inicial M0, o número de senhas em cada lugar lj pertencente a L, j = 1,..,NT, for sempre menor que n.

Uma rede de Petri limitada com n = 1 é chamada rede de Petri segura.

Numa rede de Petri limitada, o número de marcações é finito. Por outro lado, uma rede de Petri não limitada possui infinitas marcações, significando que o sistema físico correspondente é impossível de ser implementado.

b) Vivacidade

Uma rede de Petri é viva para uma marcação inicial MO, se para qualquer marcação decorrente M e para cada barra b considerada, existe uma sequência de disparos, de forma que essa barra seja habilitada.

Matematicamente,

$$\forall\, M \in \{D(MO)\},\ \forall\, b \in T,\ \exists\, Tal\ /\{\ M \xrightarrow{Tal'} M' \wedge M' \xrightarrow{b} M''\}$$

$$Tal = \{\ Tal',\ b\};\ M',\ M'' \in \{D(MO)\}$$

Um sistema físico representado por uma rede de Petri viva é livre de impasses (deadlock). Isto é, é um sistema que não possui situações conflitantes.

c) Reiniciabilidade

Uma rede de Petri é reiniciável para uma dada marcação inicial MO, se para qualquer marcação decorrente M pertencente a {D(MO)} existe uma sequência de disparo tal que faça a rede voltar à marcação inicial.

Matematicamente,

$$\forall\, M \in \{D(MO)\}\ ,\ \exists\, Tal\ /\ M \xrightarrow{Tal} MO$$

Um sistema físico, representado por rede de Petri reiniciável tem características de retornar ao seu estado inicial após a execução de uma ou mais tarefas. Esta, geralmente, se constitui uma condição necessaria para o bom funcionamento de vários sistemas.

2.3.3 - Redução de Lugares da RP.

A análise das propriedades de RP em um computador pode-se tornar bastante demorada, e eventualmente até impossível de ser efetuada. Isto ocorre porque os procedimentos destinadas a verificação das propriedades clássica da RP são obtidas através da análise das marcações atingíveis da rede. Em particular, a análise da vivacidade da rede só é possível através do cálculo exaustivo das marcações e do exame das relações entre as mesmas.

Pode-se mostrar que o número de marcações da rede cresce muito rapidamente com a complexidade da mesma [3,5,13]. O exemplo dado a seguir ilustra como o número de marcações pode crescer substancialmente mesmo para redes relativamente pequenas.

Seja a rede de Petri da Fig. 2.10a. Se o lugar l3 contém n senhas na marcação inicial é então possível que o lugar l4 tenha 2xn senhas. Esta rede tem então ao menos 2xn marcações diferentes, mas é limitada.

A rede de Petri da Fig. 2.10b é a rede da Fig 2.10a incrementada de alguns lugares e barras. Se o lugar l7 tiver n senhas inicialmente, então os lugares l3 e l6 poderão conter também n senhas. É então possível de se repetir n vezes a sequência de disparo do caso anterior. Então esta rede possui um número finito de marcações atingíveis mas este número é superior

a $n \times 2^n$. Tornando-se então, quase impossível de se analisar todas as marcações da rede.

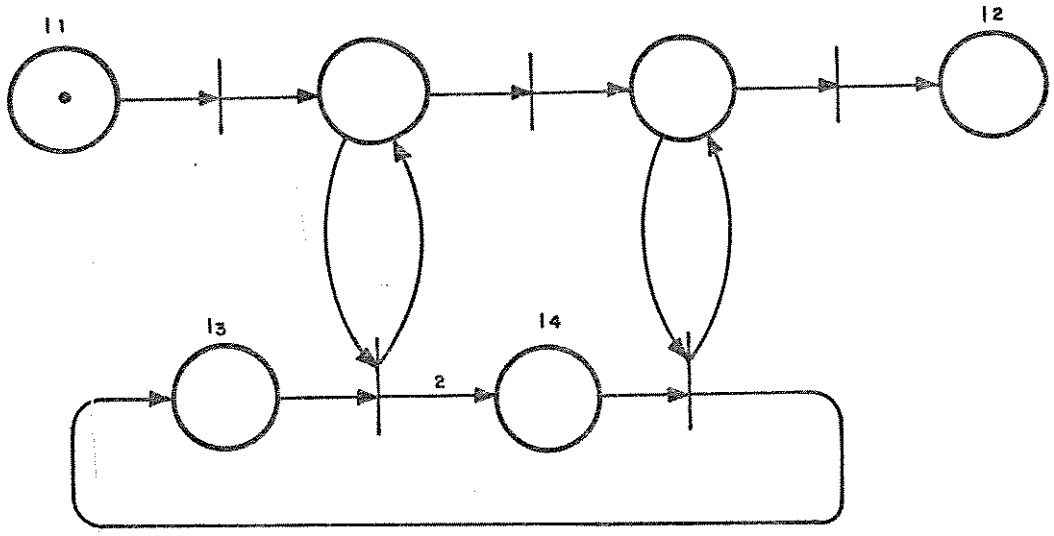


Fig. 2.10a - Exemplo de RP com grande número de marcações.

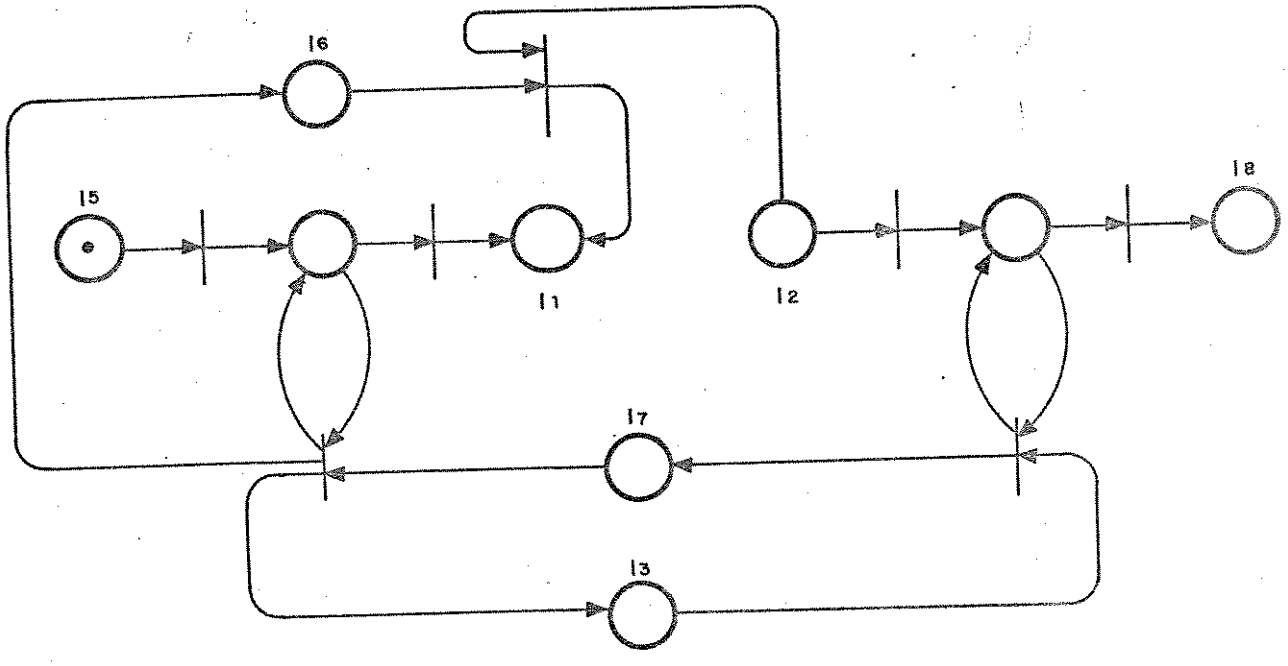


Fig. 2.10b - RP da Fig. 2.10a incrementada de alguns lugares e transições.

Uma possibilidade de análise consiste em fazer uma redução no número de marcações atingíveis, mantendo as principais propriedades da rede. Após realizar estudos de regras de simplificação evidentes, Berthelot [3] propôs um conjunto de regras que permitem eliminar certos lugares, mas sem alteração das propriedades gerais da rede.

Estas regras de redução de lugares, são destinadas a facilitar a verificação de certas propriedades da rede de Petri, proporcionando a diminuição do tempo computacional necessário para a análise de algumas redes. Entretanto, em alguns casos, a rede reduzida não carrega a mesma quantidade de informação que a rede original. Isto significa que após a aplicação das regras de redução, a relação entre os lugares e barras pode estar mudada a ponto de não ser possível localizar na rede original o que leva a não verificação de alguma propriedade, caso isto ocorra.

O objetivo da redução é então, fornecer de uma forma mais rápida, as principais características de uma rede de Petri cujo número de marcação é tão elevado que seu calculo sem a redução seria impossível.

2.3.3.1 - Substituição de Lugar

Seja a rede de Petri da Fig. 2.11a. Suponha que com o disparo da barra b1, o lugar l2 recebe uma senha. Esta senha habilitará os disparos de b2, b3 e b4. Apenas uma dessas barras pode disparar já que l2 contém apenas uma senha. Assim um dos lugares l3, l4 ou l5 receberá uma senha de acordo com a barra disparada. Por conseguinte, a marcação da rede onde o lugar l2 contém senhas é uma marcação intermediária a uma marcação onde o

lugar l2 não contém senhas. Poderíamos então considerar apenas a existência de três barras b5, b6 e b7 cujas funções de entrada fossem as de b1 e cujas funções de saída fossem respectivamente aquelas de b2, b3 e b4. Isto está representado na Fig. 2.11b.

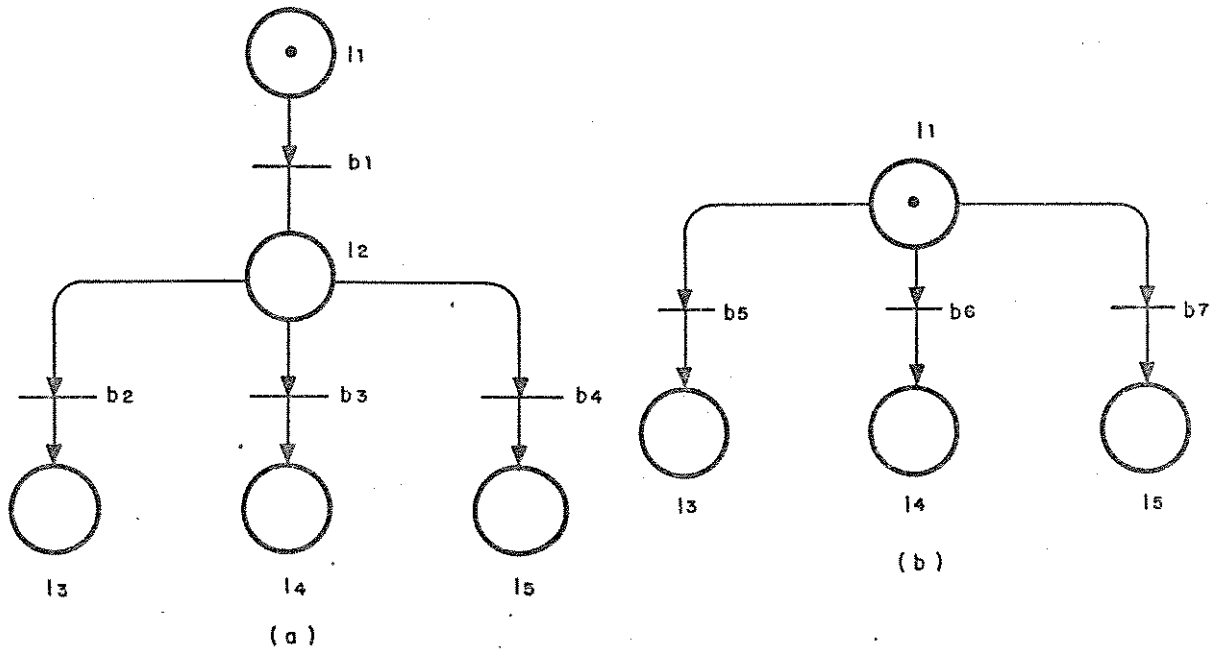


Fig 2.11 - a) Rede de Petri com lugar substituível.
b) RP do a) com lugar substituído.

Note que se l2 contém senhas na marcação inicial teremos agora três marcações iniciais, isto é, uma marcação inicial com uma senha em l3, uma outra marcação inicial com uma senha em l4 e uma outra marcação com senha em l5.

O raciocínio que foi feito para substituição do lugar l2 só é possível, quando o número de senhas contido no lugar l1 é a única condição necessária para o disparo de todas as barras que possuem o lugar l1 como entrada. Se esta condição não é

satisfeita, não é possível prever como a senha contida no lugar l1 será utilizada.

É também necessário que estas barras não sejam, ao mesmo tempo, o lugar l como lugar de entrada e lugar de saída (lugar realimentado), pois isto impossibilita novamente a previsão de como os disparos irão acontecer.

Este conjunto de considerações e ainda a necessidade de se conservar as características de vivacidade e limitabilidade levam então à definição de um lugar substituível, proposto em Berthelot [3].

2.3.3.2 - Definição de Lugar Substituível

- Seja RP = (L,T,Alfa,Beta).
- Seja l um elemento de L.
- Sejam H e F dois subconjuntos de T definidos por :

bh ∈ H <===> ∀ bh ∈ T , Beta (bh,l) ≠ 0

bf ∈ F <===> ∀ bf ∈ T , Alfa (bf,l) ≠ 0

H representa o conjunto de barras de entrada do lugar l.

F representa o conjunto de barras de saída do lugar l.

Observaçao : Nas seções onde tratar do assunto de redução de lugares, uma barra será de entrada ou de saída em relação a um lugar, diferente daquela considerada na seção 2.3.1, onde os lugares são considerados entradas e saídas em relação à uma barra.

O lugar l é dito substituível se as condições a, b, c e d abaixo forem satisfeitas.

a) Nenhuma barra pode ser ao mesmo tempo, barra de entrada e

barra de saída do lugar l.

b) ∀ bf ∈ F, Alfa (bf,l) = m, m ∈ N

∀ lq ∈ {L − l}, Alfa (bf,lq) = 0

Toda barra de saída de l tem apenas um lugar de entrada e os pesos dos arcos de saída de l são todos iguais a m.

c) ∀ bh ∈ H, existe Kh / Beta (bh,l) = Kh x m.

Todos os arcos de entrada do lugar l tem pesos múltiplos de m, e número de senhas que l contém é zero ou algum múltiplo de m.

d) ∀ bf ∈ F, existe lq ∈ {L − l} tal que Beta (bf,lq) ≠ 0.

Isto significa que ao menos uma barra de saída está ligado a algum outro lugar da rede. Esta condição garante que uma rede não limitada seja transformada em rede limitada.

Exemplo 2.2 - LUGAR SUBSTITUÍVEL.

O lugar l1, da Fig. 2.12, que tem

H = {b1, b2} - Conjunto de barras de entrada; e

F = {b3, b4, b5} - Conjunto de barras de saída

é um lugar substituível, pois, satisfaz todas as condições de um lugar substituível, conforme apresentadas abaixo.

a) Os lugares de saída das barras b3, b4 e b5 são diferentes do l1 (não realimentada).

b) As barras b3, b4 e b5 apresentam apenas um lugar de entrada, e o peso dos arcos que incidem a estas barras são iguais a m.

c) O peso do arco que incidem a l1 são k1 * m e k2 * m, onde k1 e k2 são números inteiros e positivos.

d) Todas as barras de saída apresentam pelo menos um lugar de saída.

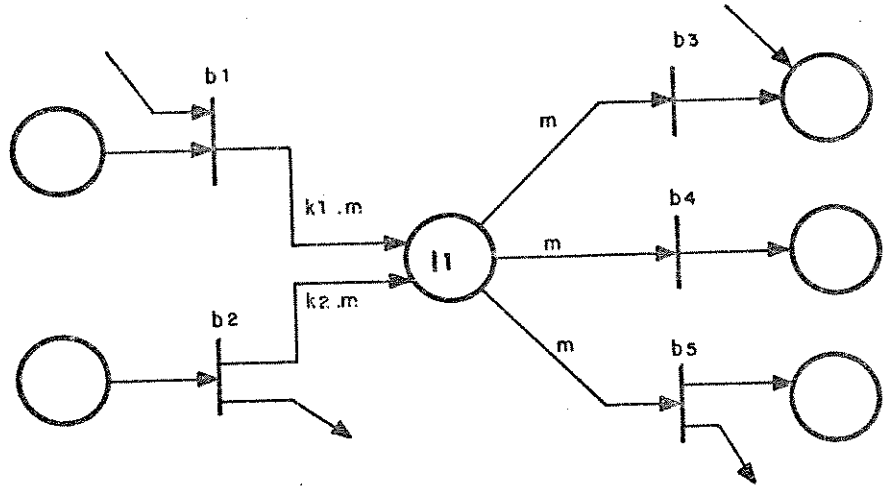


Fig. 2.12 - Exemplo de um lugar substituível.

2.4 - Conclusão

Neste capítulo foram apresentados as regras e procedimentos básicos da linguagem SDL, utilizada na representação gráfica de processos concorrentes.

Em seguida, foram apresentadas as principais definições e propriedades de rede de Petri, e uma técnica de redução de lugares de rede de Petri.

CAPÍTULO 3

REGRAS DE CONVERSÃO E IMPLEMENTAÇÃO DO PROGRAMA

3.1 - Introdução

Neste capítulo são apresentadas inicialmente as regras para realizar a conversão de um protocolo especificado em SDL para a sua representação em rede de Petri [6,7].

Em seguida, são apresentados um método para representar as matrizes esparsas e algorítmos implementados para verificar as propriedades da Rede de Petri. São apresentados também, os programas de computador desenvolvidos para realizar automàticamente a conversão do protocolo em SDL para rede de Petri, verificar as propriedades da rede de Petri e documentar os resultados da análise.

3.2 - Regras para Conversão de SDL para RP

A seguir são apresentadas as regras para a conversão da representação gráfica em SDL de um protocolo para sua equivalente representação em RP. Estas regras não se aplicam a protocolos com temporizadores.

Para a conversão é definido o termo ramo como sendo o caminho entre dois símbolos de estado consecutivos, mas que contenha obrigatòriamente o símbolo de entrada entre estes estados.

No caso em que, entre dois estados exista um símbolo de decisão, cada caminho diferente entre dois estados deve ser considerado como um ramo.

As regras para a conversão de SDL para RP, são:

1 - Todos os símbolos SDL de estado que pertencem a um mesmo processo e com nomes iguais corresponde a um único lugar na RP equivalente.

2 - Um símbolo SDL de entrada (saída) no diagrama SDL de um processo, com seu correspondente símbolo de saída (entrada) no diagrama SDL de um outro processo será representado por um mesmo lugar na RP equivalente.

3 - Cada ramo SDL entre dois símbolos SDL de estado corresponde a uma barra de transição na RP equivalente.

- O símbolo SDL de estado no qual o ramo SDL se inicia corresponde a um lugar de entrada da barra de transição.

- O símbolo SDL de entrada é também um lugar de entrada da barra de transição.

- O símbolo SDL de estado no qual o ramo SDL termina é um lugar de saída da barra de transição.

- O símbolo SDL de saída do ramo SDL, também corresponde a um lugar de saída da barra de transição.

4 - Ficará sem representação na RP equivalente :

- Todo símbolo SDL de tarefa.

- Todo símbolo SDL de entrada ou de saída que corresponda a um processo não especificado por diagrama SDL.

Exemplo 3.1 - APLICAÇÃO DAS REGRAS DE CONVERSÃO DE SDL PARA REDE DE PETRI

O protocolo, utilizado na rede digital de serviços integrados - RDSI para suspensão e religação de chamadas

telefônicas ou de dados apresentado na diagrama SDL da Fig. 2.3, é representado por dois processos. Estes processos pertencem a blocos diferentes, pois se interagem através de troca de sinais externos.

O primeiro Bloco funcional contém o processo 1, que representa o Lado Usuário, é formado por 6 ramos (transições) enumeradas de 1 a 6 e 17 símbolos enumeradas de 1 a 14 (o símbolo 1 é repetido 2 vezes e o símbolo 7 uma vez); e o segundo bloco funcional contendo o processo 2, que representa o Lado Rede, é formado por 7 ramos enumeradas de 11 a 17 e 19 símbolos enumeradas de 21 a 36.

Estes números (enumerações) de blocos funcionais, processos, ramos e de símbolos serão utilizadas respectivamente como código do bloco funcional, do processo, do ramo e do símbolo para a sua identificação no diagrama.

Neste protocolo, pode-se aplicar as regras de conversão apresentadas na seção 3.2 da seguinte forma :
Inicialmente, converte-se cada um dos ramos do processo 1 em ordem crescente dos códigos de ramos para respectiva representação em rede de Petri e em seguida faz o mesmo com os ramos do processo 2, também obedecendo a ordem crescente dos códigos do ramo.

A rede de Petri equivalente obtida após esta conversão é apresentada na Fig 3.1 contendo 13 barras e 14 lugares.

Nesta rede de Petri podemos observar que :
- A quantidade de barras é a mesma da quantidade de ramos e os respectivo codigo é o mesmo do ramo correspondente.
- A quantidade de lugares é menor do que quantidade de símbolos e

o codigo de lugares estão em ordem crescente, sem ter relação direta com os codigo dos símbolos.

- A quantidade de senhas na marcação inicial é igual a quantidade de processos que contém no diagrama SDL e estão distribuídas nos lugares que representam o estado inicial dos processos.

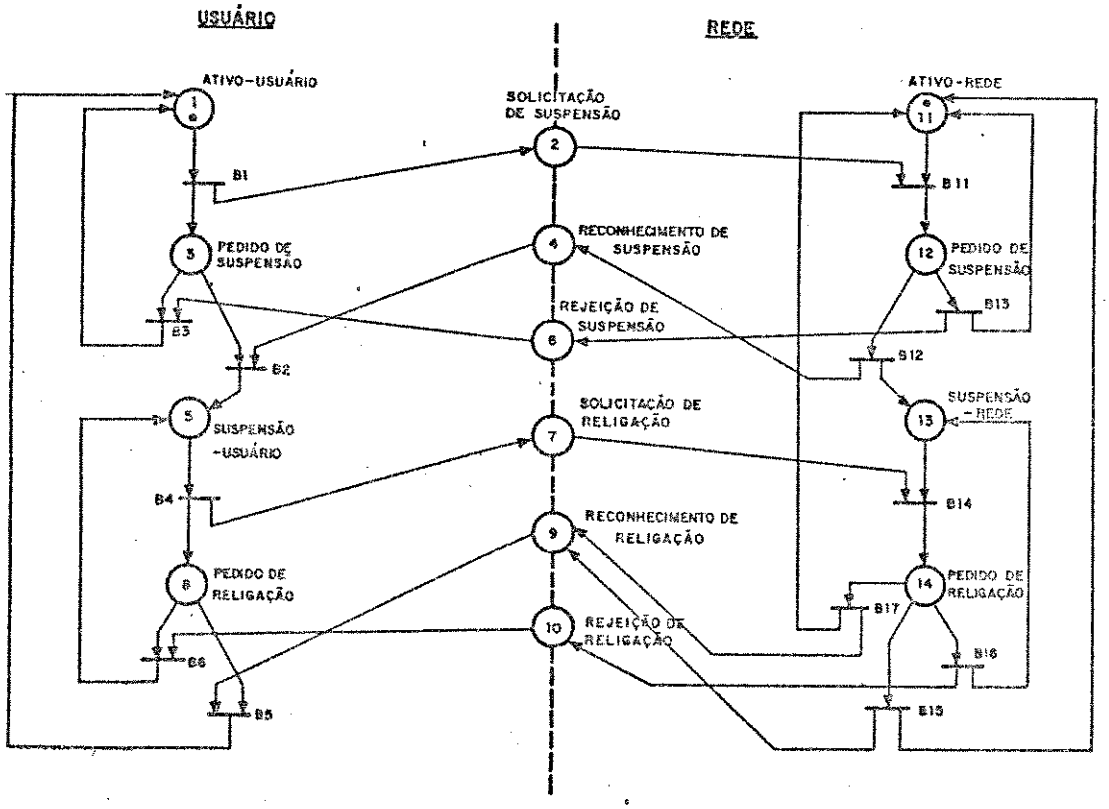


Fig. 3.1 - Rede de Petri equivalente do protocolo da Fig 2.3

3.3 - Algorítmos Implementados

Nesta Seção, são mostradas inicialmente, que as matrizes utilizadas para a representação da matriz de incidência, tabela de marcações, etc., apresentadas no capítulo 2 são matrizes esparsas, portanto necessitam da técnica de alocação dinâmica para utilização eficiente da memória do computador.

Em seguida, são apresentados os algorítmos utilizados na implementação da técnica de redução de lugares e das análises da limitabilidade, da reiniciabilidade e da vivacidade de uma RP.

3.3.1 - Matrizes Esparsas e suas Representações

No capítulo 2, foram apresentadas quatro matrizes conhecidas como matriz de incidência direta, matriz de incidência reversa, tabela de marcações e máquina de senha que são utilizadas para a análise de rede de Petri.

Em particular, na análise de rede de Petri correspondente a um protocolo, verificou-se que a maior parte das posiçoes destas matrizes não armazenavam informações relevantes à análise, ou seja, são matrizes esparsas. Portanto um ferramental para auxiliar na validação de protocolos implementado através de manipulações destas matrizes não teria um bom desempenho.

A Fig. 3.2a, Fig. 3.2b, Fig. 3.2c e Fig. 3.2d mostram respectivamente a matriz de incidência direta, matriz de incidência reversa, tabela de marcações e máquina de senhas da rede de Petri da Fig. 3.1.

Barras\Lugares	1	2	3	4	5	6	7	8	9	10	11	12	13	14
B1	1	–	–	–	–	–	–	–	–	–	–	–	–	–
B2	–	–	1	1	–	–	–	–	–	–	–	–	–	–
B3	–	–	1	–	–	1	–	–	–	–	–	–	–	–
B4	–	–	–	–	1	–	–	–	–	–	–	–	–	–
B5	–	–	–	–	–	–	–	1	1	–	–	–	–	–
B6	–	–	–	–	–	–	–	1	–	1	–	–	–	–
B11	–	1	–	–	–	–	–	–	–	–	1	–	–	–
B12	–	–	–	–	–	–	–	–	–	–	–	1	–	–
B13	–	–	–	–	–	–	–	–	–	–	–	1	–	–
B14	–	–	–	–	–	–	1	–	–	–	–	–	1	–
B15	–	–	–	–	–	–	–	–	–	–	–	–	–	1
B16	–	–	–	–	–	–	–	–	–	–	–	–	–	1
B17	–	–	–	–	–	–	–	–	–	–	–	–	–	1

(a)

Barras\Lugares	1	2	3	4	5	6	7	8	9	10	11	12	13	14
B1	–	1	1	–	–	–	–	–	–	–	–	–	–	–
B2	–	–	–	–	1	–	–	–	–	–	–	–	–	–
B3	1	–	–	–	–	–	–	–	–	–	–	–	–	–
B4	–	–	–	–	–	–	1	1	–	–	–	–	–	–
B5	1	–	–	–	–	–	–	–	–	–	–	–	–	–
B6	–	–	–	–	–	1	–	–	–	–	–	–	–	–
B11	–	–	–	–	–	–	–	–	–	–	–	1	–	–
B12	–	–	–	1	–	–	–	–	–	–	–	–	1	–
B13	–	–	–	–	–	1	–	–	–	–	1	–	–	–
B14	–	–	–	–	–	–	–	–	–	–	–	–	–	1
B15	–	–	–	–	–	–	–	–	1	–	1	–	–	–
B16	–	–	–	–	–	–	–	–	–	1	–	–	1	–
B17	–	–	–	–	–	–	–	–	1	–	1	–	–	–

(b)

Marcações\Lugares

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
M0	1	–	–	–	–	–	–	–	–	–	1	–	–	–
M1	–	1	1	–	–	–	–	–	–	–	1	–	–	–
M2	–	–	1	–	–	–	–	–	–	–	–	1	–	–
M3	–	–	1	1	–	–	–	–	–	–	–	–	1	–
M4	–	–	1	–	–	1	–	–	–	–	1	–	–	–
M5	–	–	–	–	1	–	–	–	–	–	–	–	1	–
M6	–	–	–	–	–	–	1	1	–	–	–	–	1	–
M7	–	–	–	–	–	–	–	1	–	–	–	–	–	1
M8	–	–	–	–	–	–	–	1	1	–	1	–	–	–
M9	–	–	–	–	–	–	–	1	–	1	–	–	1	–

(c)

Marcações\Barras

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
M0	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
M1	–	–	–	–	–	–	–	–	–	–	2	–	–	–	–	–	–
M2	–	–	–	–	–	–	–	–	–	–	–	3	4	–	–	–	–
M3	–	5	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
M4	–	–	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–
M5	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–
M6	–	–	–	–	–	–	–	–	–	–	–	–	–	7	–	–	–
M7	–	–	–	–	–	–	–	–	–	–	–	–	–	–	8	9	8
M8	–	–	–	–	0	–	–	–	–	–	–	–	–	–	–	–	–
M9	–	–	–	–	–	5	–	–	–	–	–	–	–	–	–	–	–

(d)

Fig. 3.2 – (a) Matriz de incidencia direta
(b) Matriz de incidencia reversa
(c) Tabela de marcações
(d) Maquina de senha
da rede de Petri da Fig. 3.1

Segundo Collins [10], é conveniente o uso de alocação dinâmica de memória de computador para representar uma matriz esparsa quando tem pelo menos 80 porcentos das posições da matriz sub-utilizadas.

A seguir é apresentada a conveniência do uso de alocação dinâmica de memória no programa de computador que foi implementado com o objetivo principal de análise de RP correspondente de um protocolo de comunicação.

a) Matriz de Incidência Direta (MID)

Como foi visto na regra de conversão, cada barra da RP corresponde a um ramo da SDL. Logo, pela definição da SDL, cada barra pode ter no máximo dois lugares de entrada, onde um lugar corresponde o estado e outro ao sinal de chegada.

Portanto, para cada linha de MID, apenas duas posições contêm dados referentes a lugar de entrada.

Como exemplo, podemos calcular o fator ocupacional da MID da Fig. 3.2a

Fat. ocupacional = (19 posições x 100)/(13 linhas x 14 colunas) = 10,4 %

b) Matriz de Incidência Reversa (MIR).

Da mesma forma, conforme a definição da SDL, cada barra deve ter um lugar de saída que corresponde ao proximo estado atingido. E ainda, podemos considerar que para cada mudança de estado, um processo envia um sinal em média para outro processo. Logo, em média, cada barra terá dois lugares de saída, isto significa que apenas duas posições de MIR contêm dados referentes a lugares de saída.

O fator ocupacional da MIR da Fig. 3.2b será dado por

Fat. ocup. = (20 posições x 100)/(13 linhas x 14 colunas) = 11 %.

c) Tabela de Marcações (TM)

Na RP equivalente de um protocolo, podemos classificar dois conjuntos de lugares da RP: o primeiro corresponde aos estados dos processos que compõem o diagrama SDL e o segundo aos sinais trocadas entre os processos.

Em todas as marcações existirão quantidade de senhas equivalente ao mesmo número de processos distribuídos um em cada lugar no primeiro conjunto de lugares. E conforme a consideração feita no caso da MIR, pode existir em média o mesmo número de senhas distribuídas um em cada lugar no segundo conjunto de lugares (rede limitada). Isto é, num protocolo representado por n

processos, em cada marcação de RP teremos em média 2n senhas distribuídas uma em cada lugar.

O fator ocupacional da TM da Fig. 3.2c pode ser dado por

Fat. ocup = (26 posições x 100)/(14 colunas x 10 linhas)

= 19 %

d) Máquina de senha (MS)

Analisando os resultados de simulações de vários protocolos, verificou-se que a quantidade média de barras habilitadas para o disparo em cada marcação pode ser considerada como igual ao número de processos no diagrama. Logo se tiver n processos no diagrama, temos em média n barras disparáveis em cada marcação.

O fator ocupacional da MS da Fig. 3.2d pode ser calculada por

Fat. ocup. = (13 posições x 100)/(14 colunas x 10 linhas)

= 10 %

Para armazenar as informações das matrizes esparsas anteriormente citadas na forma de alocação dinâmica de memoria, foi utilizada a técnica de filas, onde cada posição da fila permite um acesso direto a "record" ou bloco com um apontador (recurso da linguagem pascal).

Este bloco apresenta a seguinte estrutura :

```
VETOR = ^BLOCO;

BLOCO = RECORD
          CODIGO, PESO : INTEGER;
          PROX   :  VETOR
        END;
```

Este tipo de declaração permite o encadeamento de vários blocos sequencialmente, interligados através da variável PROX. Esta variável do último bloco deve estar sempre apontando para uma posição nula que em linguagem Pascal é conhecido como "nil".

A Fig. 3.3 mostra este encadeamento dos blocos na forma de figura.

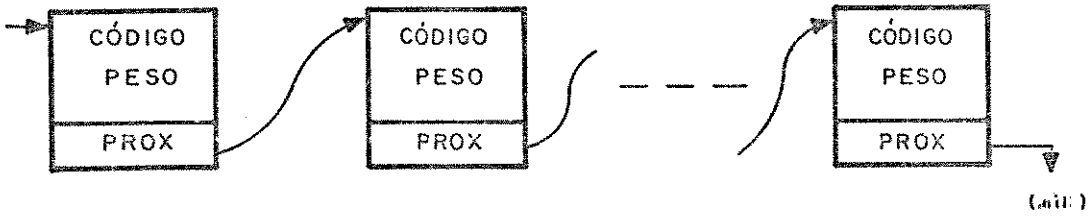


Fig. 3.3. - Exemplo de encadeamento de blocos.

A seguir serão apresentadas as filas criadas para cada matriz e as informações armazenadas em cada variavel do bloco.

a) Matriz de Incidência Direta

Para esta matriz, foi declarada a variável M_INC com a seguinte estrutura : M_INC = array[1..Max_bar] of VETOR;

Esta variável permite tratar uma rede de Petri que tenha até Max_bar barras e possibilita o encadeamento de blocos em cada uma das Max_bar posições. O acesso a uma barra bj, é conseguido diretamente por M_INC[bj]. E cada bloco encadeado nesta posição da fila armazena seguintes informações:

CODIGO : Codigo do lugar lj que pertence ao conjunto de lugares de entrada da barra bj.

PESO : Peso do arco orientado que liga o lugar lj a barra bj.

Como exemplo, a Fig. 3.4 mostra o encadeamento dos blocos

das barras B6 e B7 fa Fig. 3.2a.

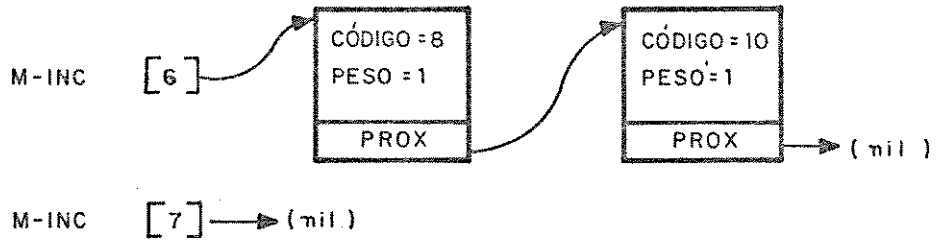


Fig. 3.4 - Exemplo de uma parte da matriz incidencia direta.

b) Matriz de Incidência Reversa

Para esta matriz foi declarada a variavél M_REV com a mesma estrutura e dimensão do variável M_INC. E nos blocos encadeados na posição bj, estão armazenadas as seguintes informações:

CODIGO : O codigo do lugar lj que é um dos lugares de saída da barra bj.

PESO : Peso do arco de incidência que sai da barra bj e chega a lugar lj.

c) Tabela de Marcações

Neste caso, foi declarada a variável TAB_MARC com seguinte estrutura : TAB_MARC : array[1..Max_marc] of VETOR

Esta fila permite analisar uma rede de Petri que tenha até Max_marc marcações. O acesso à marcação Mj, é conseguido diretamente por TAB_MARC[Mj] e cada bloco encadeado nesta posição armazena seguinte informações :

CODIGO : Codigo do lugar li que contém pelo menos uma senha na marcação Mj.

Peso : Quantidade de senhas que contém o lugar li da marcação Mj.

d) Máquina de Senha

Para esta matriz foi declarada a variável MAC_SENHA, que é da mesma estrutura e dimensão da TAB_MARC. E em cada bloco encadeado, na posição Mj, estão armazenadas as seguintes informações:

CODIGO : Codigo da barra bj habilitada para disparo na marcação Mj.

Peso : Codigo da proxima marcação atingida após o disparo da barra bj na marcação Mj.

3.3.2 - Redução de Lugar

Na técnica de redução é feita a substituição de um lugar (l) com suas barras de entrada (H) e barras de saída (F) por um conjunto de novas barras, de forma que as marcações obtidas pelo disparo de novas barras sejam identicas aos disparos das diferentes combinações de bh e bf [13].

No algorítmo de redução implementado, um lugar l é substituível se satisfizer as condições a, b, c, d e e abaixo citadas.

a) O lugar l não possi senhas na marcação inicial.

b) O lugar l não deve possuir barra b tal que b ∈ H e b ∈ F (O lugar l não deve ser realimentado)

c) Todas as barras pertencentes a F deve ter apenas um lugar de entrada (o próprio l)

d) Os pesos das arcos que entram ou saem do lugar l devem ser iguais a 1 (m = 1).

e) O lugar l deve ter pelo menos uma barra de saída.

Através da condição d, o número de barras criadas (NTC) para

substituir o conjunto I, H e F é igual a n x p, onde n é a quantidade das barras de saída do lugar I (n = dim(F)) e p é a quantidade das barras de entrada do lugar I (p = dim(H)).

Demonstração :

$$NTC = \sum_{i=1}^{p} FUNC(Ki,n) = \sum_{i=1}^{p} \sum_{j=1}^{n} Cn,j \; func(Ki-j,j)$$

mas Ki = 1, pois todos os arcos tem peso = 1; e

func(0,1) = 1 e func(i,j) = 0 se i,j ≠ 0 e 1

logo temos que

$$NTC = \sum_{i=1}^{p} Cn,1 = \sum_{i=1}^{p} n = np$$

A Fig. 3.5 apresenta o fluxograma do algoritmo de redução implementado. Neste algorítmo, a análise de redução de lugar é feita em ordem crescente dos códigos dos lugares que pertencem a RP. Para cada lugar I em análise, são criadas duas filas denominadas de H e F que armazenam respectivamente os código das barras de entrada e das barras de saída do lugar I.

Se o lugar I não é redutível, então as informações das filas F e H são apagadas e reinicia a análise com outro lugar Ij que pertence a RP.

Caso contrário, são criadas n x p novas barras com códigos diferentes das barras que pertencem à RP neste instante.

Para a explicação da técnica utilizada para criar uma barra b, considere duas barras bh e bf que são respectivamente barra de entrada e de saída do lugar I.

Para a nova barra b, os lugares de entrada são os mesmos da barra bh e os lugares de saída são a soma dos lugares de saída da

barra bf e da barra bh com o lugar l excluído.

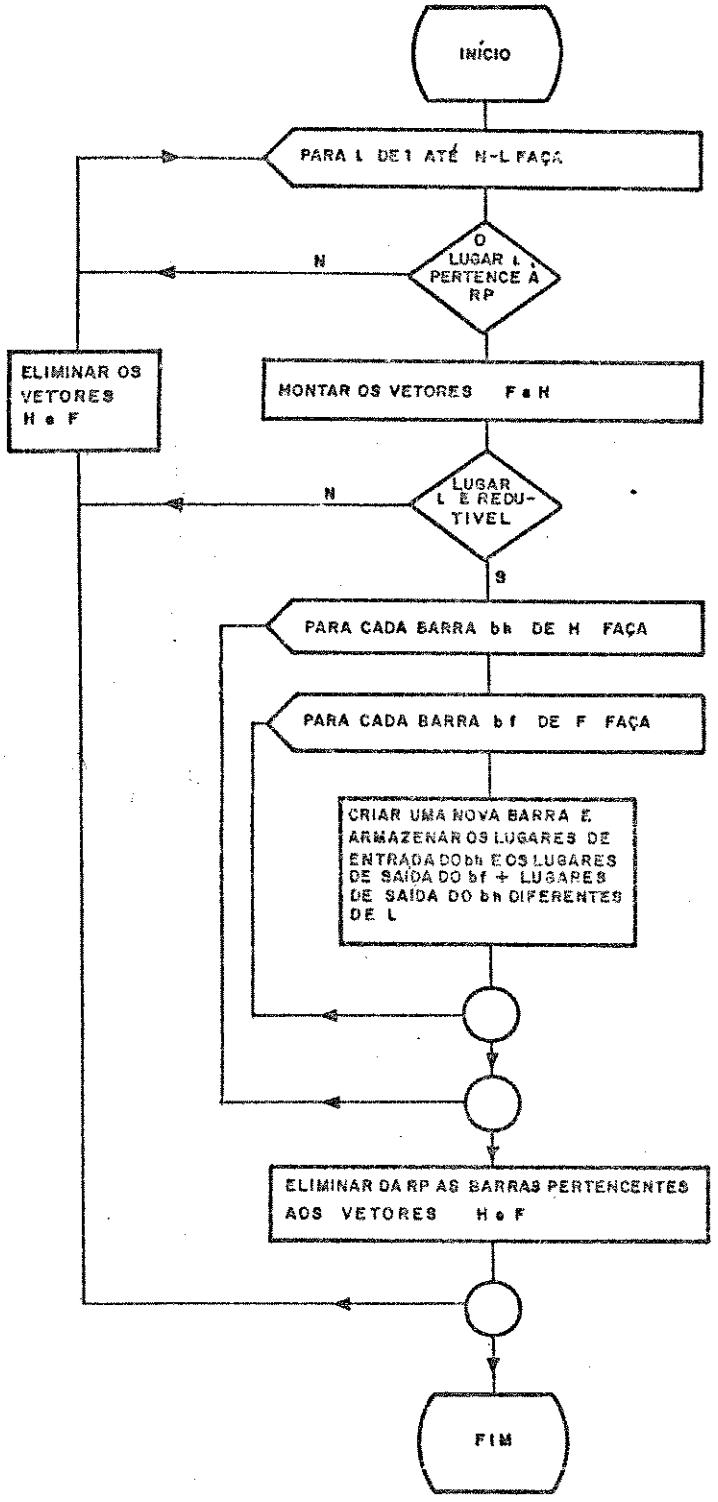


Fig. 3.5 - Fluxograma do Algorítmo de Redução de Lugar.

Após criar np novas barras, são retiradas da RP as barras de entrada e de saída do lugar l. Consequentemente o lugar l será suprimido da RP. Em seguida, as informações das filas F e H são apagadas e reinicia análise para um outro lugar da RP.

Exemplo 3.2 - EXEMPLO DE REDUÇÃO DE LUGAR

Será utilizado o exemplo de rede de Petri da Fig. 3.1.

Observando a Fig. 3.1, podemos verificar que os lugares l1 e l11 não são redutíveis por conter as senhas de marcação inicial; e os lugares l2, l3, l4, l6, l7, l8, l9, l10 e l13 não são redutíveis porque todos estes lugares tem as barras de saída sendo acessadas por dois lugares diferentes.

Finalmente, os lugares, l5, l12 e l14 são redutíveis. A seguir são mostradas o procedimento seguido para a redução destes lugares.

-> Redução do lugar l5

Para este lugar são montadas as filas H e F com as seguintes informações :

H = {b2, b6} p = dim{H} = 2
F = {b4} n = dim{F} = 1

Logo, o conjunto l5, b2, b6 e b4 é substituído por np = 2 novas barras.

O código da primeira nova barra criada é 7, pois é o menor código possível das barras e que não está sendo utilizado na RP neste instante.

Os lugares de entrada da b7 são l3 e l4, os mesmos da barra b2. E os lugares de saída são l7 e l8.

O código da segunda barra criada é b8. Esta barra tem

lugares l8 e l10 de entrada, e os lugares de saída são l7 e l8.

Após criar estas duas barras, em seguida as barras b2, b4 e b6 são eliminadas da RP, zerando a M_INC e M_REV de cada um destes códigos.

-> Redução do lugar l12.

Para este lugar temos:

H = {b11} p = 1

F = {b12, b13} n = 2

Neste caso tambeém o conjunto l12, b11, b12 e b13 é substituído por duas barras.

Os códigos das novas barras criadas serão b2 e b4, pois as barras com estes códigos não estão pertencendo a RP neste instante.

A barra b2 tem os lugares l2 e l11 de entrada e de saída l4 e l13. E os lugares de entrada de b4 são l2 e l11, e os lugares de saída são l6 e l11.

Armazenadas estas informações na M_INC e M_REV, as barras b11, b12 e b13 são eliminadas da RP e consequentemente o lugar l12.

-> Redução do lugar l14

para este lugar temos:

H = {b14} p = 1

F = {b15, b16, b17} n = 3

Neste caso o conjunto é substituído por 3 novas barras, cujas os codigos são b6, b9 e b10.

Os lugares de entrada das novas barras são l7 e l13, e os lugares de saída da b6 são l9 e l11, da b9 são l10 e l13 e da

b10 são I9 e I11.

Finalmente, eliminadas as barras b14, b15, b16 e b17 temos a rede de Petri com lugares reduzidos. Esta RP é apresentada na Fig. 3.6.

Analisando-se as redes de Petri das Fig. 3.1 e 3.6, verificou-se que ambas as redes satisfazem as propriedade de limitabilidade, vivacidade e reiniciabilidade. No entanto, a primeira apresentou dez marcações e a segunda, apenas sete. Portanto, a redução de lugar é uma técnica que traz uma melhoria considerável na análise, pois, além de reduzir a quantidade de marcações, não altera as propriedades da rede.

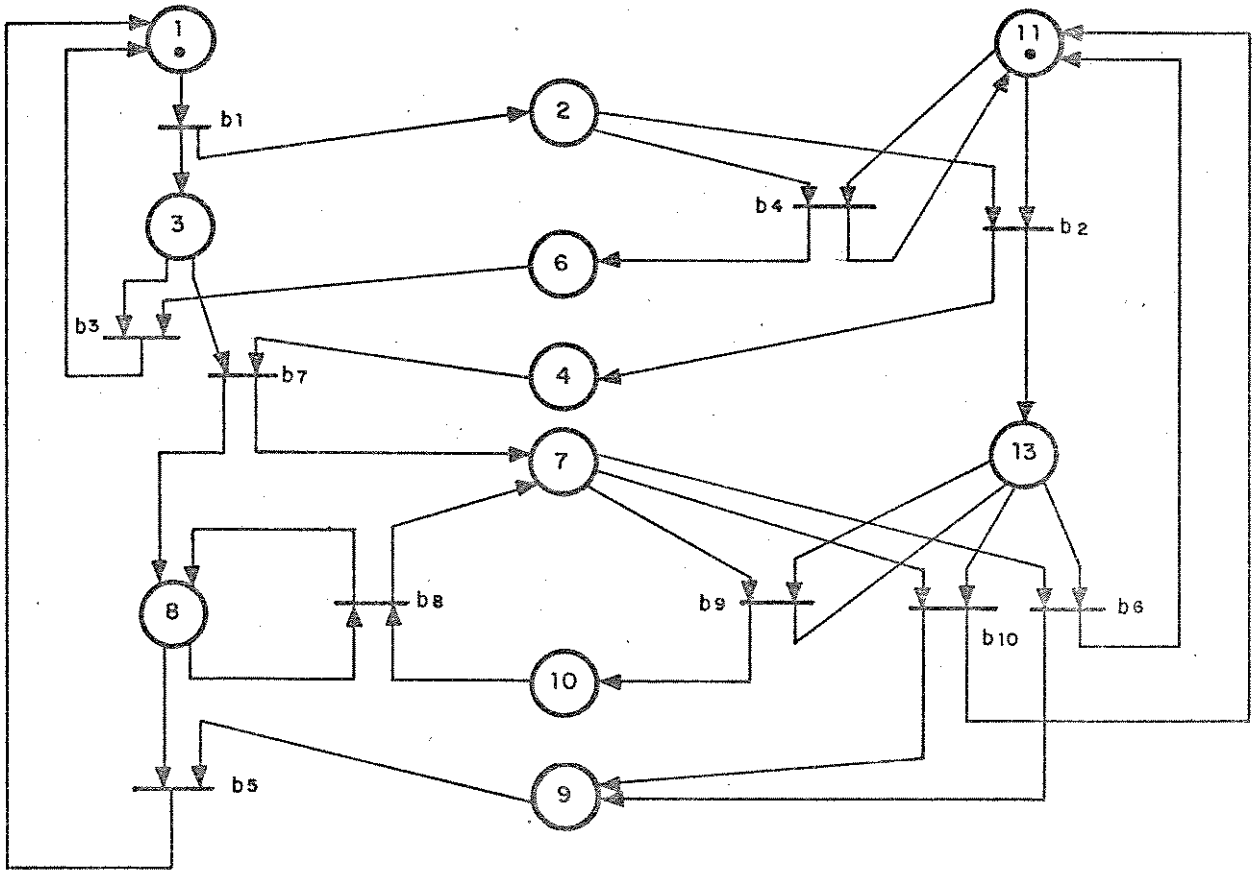


Fig 3.6 - Rede de Petri da Fig. 3.1 com lugares reduzidos.

3.3.3 - Limitabilidade

Em uma rede de Petri limitada o número de senhas em qualquer lugar, para qualquer marcação, não deve ultrapassar o valor "n" estabelecido a-priori.

3.3.3.1 - Técnica de Representação das Marcações

Uma das representações das marcações é a máquina de senha na forma matricial apresentada anteriormente (MS). Outra representação é a chamada árvore de marcações acessíveis (reachability tree) proposta por Peterson [2].

A segunda representação usa o conceito de árvore para calcular todas as marcações acessíveis, à partir, de uma dada marcação inicial. Nesta árvore, as marcações representam os nós da árvore. Os nós são interligados por arcos orientados rotulados com a barra de transição que levou àquele nó.

A Fig 3.8 mostra uma árvore de marcações de uma parte das marcações acessíveis da rede de Petri da Fig. 3.7 com marcação inicial MO = (1,0,0).

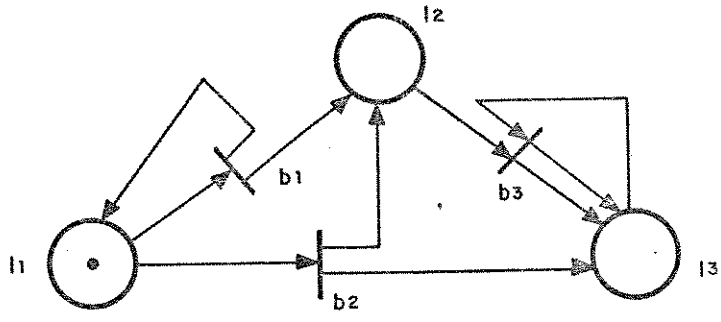


Fig 3.7 - Exemplo de RP não limitada.

Esta RP apresenta infinitas marcações, pois à partir de MO,

cada disparo de b1 que acrescenta uma senha no lugar l2 e em respectiva marcação atingida a barra b1 continua habilitada a disparar novamente. Este caso é representado pelas marcações {M0, M1, M3, M6}.

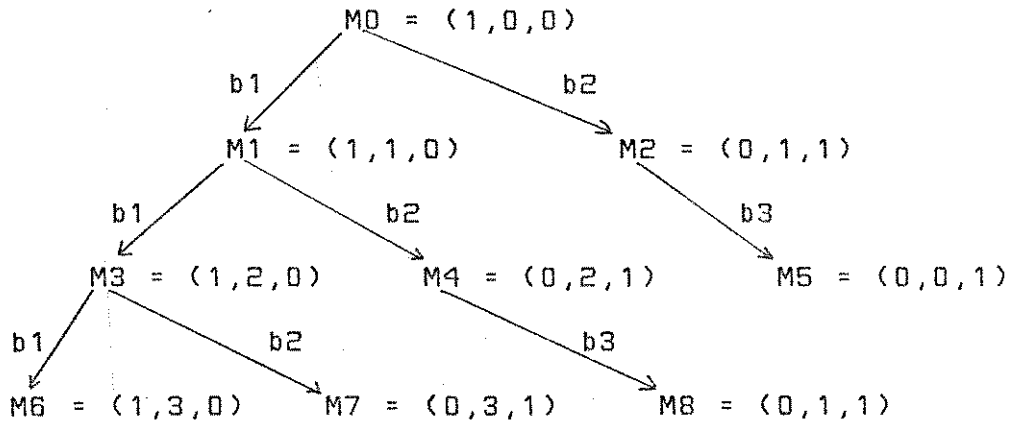


Fig 3.8 - Elaboração de árvore de marcações

Na Fig. 3.8 nota-se, também, que por exemplo a marcação (0,1,1) pode ser decorrente da marcação inicial pelo disparo de b2 ou da M4 pelo disparo de b3.

Na árvore de marcações, a partir de cada nó são definidos novos nós na árvore para as marcações que resultam dos disparos das barras. Se esse procedimento for repetido sucessivamente para todos os nós, serão produzidas as marcações acessíveis à partir de uma dada marcação inicial M0.

Neste processo de elaboração da árvore, podem ocorrer um número infinito de novas marcações, por que as marcações repetidas também são inseridas na árvore.

Desse modo para limitar a quantidade de marcações a um número finito, deve-se limitar as novas marcações introduzidas em cada estágio. Isto é feito suprimindo na árvore, as marcações que

já apareceram. Essas marcações não precisam ser consideradas, pois suas extensões já foram consideradas quando de sua primeira aparição.

Uma outra classe de marcações que deve ser analisada é aquela para as quais o número de senhas, em um ou mais lugar cresce indefinidamente. Nesses casos, o número de marcações tende a infinito.

Para limitar o número de marcações a um número finito de marcações, uma alternativa, é a de se representar as quantidade de senhas com um símbolo especial w, em todos os lugares para os quais os números de senhas crescem indefinidamente. Isto é, para uma constante a, define-se

$$w + a = w$$
$$w - a = w$$
$$a < w$$
$$w <= w$$

Desse modo, a árvore de marcações acessíveis, do exemplo da Fig 3.8 ficará

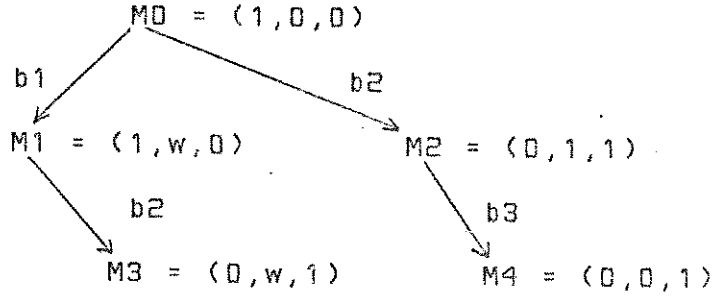


Fig 3.9 - Árvore de marcações acessíveis para a rede da Fig.3.7 com símbolo w

Com as restrições discutidas acima, pode-se provar que a árvore de marcações acessíveis é finita (Peterson).

3.3.3.2 - Definição : Ramo Ascendente

O ramo ascendente de uma marcação M é o conjunto de marcações na árvore de marcações acessíveis, que gera a marcação M. Por exemplo, seja a árvore de marcação mostrada na Fig. 3.8.

O ramo ascendente da marcação M6, é conjunto de marcações { M3, M1, M0 }.

O ramo ascendente de M8 é o conjunto {M4, M1, M0} e assim por diante.

3.3.3.3 - Teorema de Limitabilidade

Sejam as marcações M e M' ∈ {D(M0)}. Se a marcação M' é superior à M (M' > M) e M pertence ao ramo ascendente de M' , então a rede de Petri é não limitada [4].

Neste caso, o número de marcações tende a infinito, pois o número de senhas em um ou mais lugares cresce indefinidamente. O número de senhas destes lugares é representado pelo código especial w.

3.3.3.4 - Implementação

Quando a rede é não limitada, o número de marcações é infinito. Assim, não seria possivel calcular todas as marcações acessíveis dessa rede, em um computador. Desse modo, para cada marcação obtida, é verificada a limitabilidade da rede através do teorema da limitabilide.

O fluxograma de algorítmo implementado é mostrado na Fig. 3.10. Este algorítmo permite além da verificação da propriedade de limitabilidade, a construção simultânea das tabelas de marcações e de senhas.

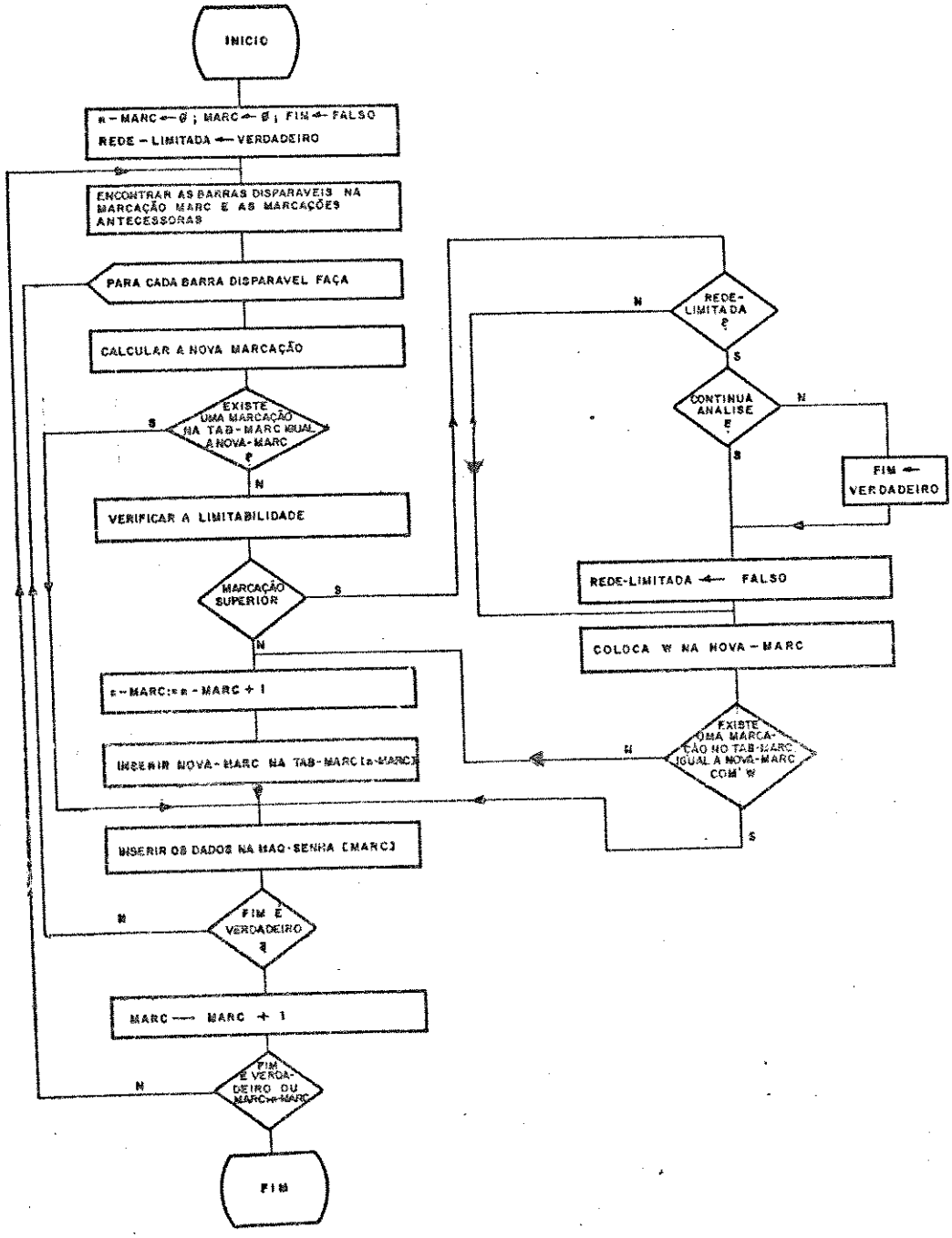


Fig 3.10 – Fluxograma para construção de tabela de marcações e a máquina de senhas, contendo a análise de limitabilidade da RP.

3.3.4 - Reiniciabilidade

Uma rede de Petri é reiniciável se para cada marcação M ∈ {D(M0)}, existir uma sequência de disparo que a leve de volta à marcação M0.

Um algorítmo para verificação desta propriedade pode ser implementado com o auxílio da máquina de senhas. Será utilizado para apresentação deste algorítmo, o exemplo de rede de Petri da Fig. 3.1, cuja tabelas de marcações e máquina de senhas para a marcação inicial M0 = (I1 = 1, I11 = 1), estão representadas nas Figs. 3.2c e 3.2d respectivamente.

Observando a Fig. 3.2d, pode-se verificar que para a marcação M4, a barra de transição habilitada a disparar é b3. O disparo de b3 leva à marcação M0, que é a marcação inicial. Assim, pode-se concluir que existe uma sequencia de disparos que partindo de M4 leva de volta à marcação inicial. Logo, a marcação M4 é reiniciável. Sob este mesmo procedimento pode-se constatar que a marcação M8 é reiniciável, pois, a partir desta marcação o disparo de b5 leva à marcação inicial.

Desse modo, nesse exemplo têm-se que as marcações M4 e M8 são reiniciáveis. Para as outras marcações, basta verificar se há disparos que levam a um das marcações reiniciáveis. Neste caso estas marcações são M4, M8 e a própria marcação inicial M0. Neste exemplo pode-se notar que as marcações M2 e M7 são reiniciáveis, pois existe uma sequência de disparos que levam respectivamente a marcação M4 e M8 que são reiniciáveis.

Da mesma forma, pode-se verificar que as marcações M1 e M6 são reiniciáveis, pois existe um sequência de disparos que levam respectivamente à marcação M2 e M7.

Utilizando-se esse procedimento, pode-se verificar a propriedade de reiniciabilidade da rede. No exemplo dado, a rede é reiniciável, pois as marcações M3, M5 e M9 também apresentam uma sequência de disparos para retornar a marcação inicial.

A Fig 3.11 mostra o Fluxograma de um algorítmo para verificação dessa propriedade utilisando-se a máquina de senha.

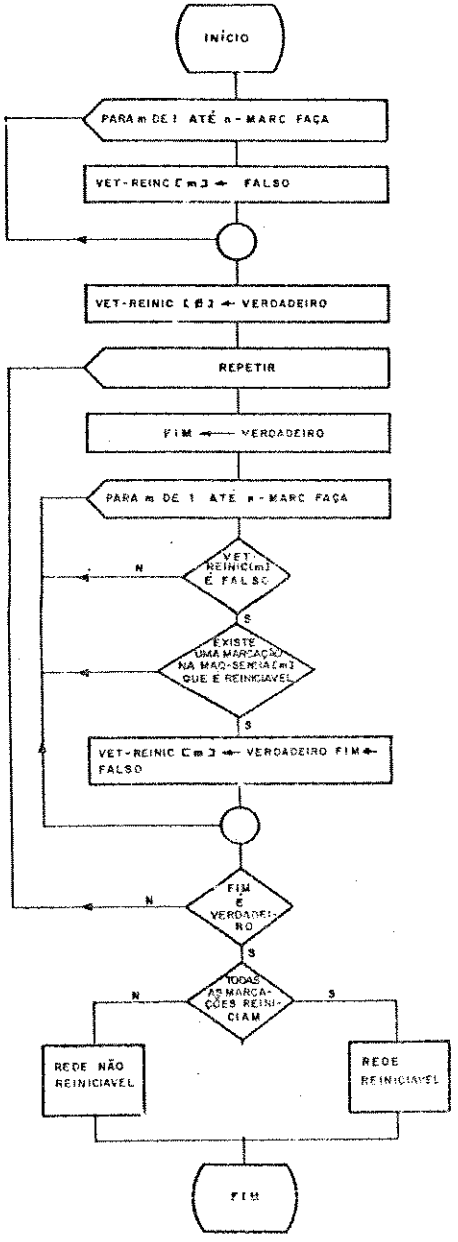


Fig. 3.11 - Fluxograma para verificação de reiniciabilidade.

3.3.5 - Vivacidade

Uma barra bi é viva se à partir de uma marcação qualquer for possível atingir uma marcação para a qual a barra bi é habilitada. Uma rede de Petri é viva se todas as barras são vivas.

Para verificação dessa propriedade, as redes de Petri podem ser divididas em dois casos : redes reiniciáveis e não reiniciáveis.

a) Redes de Petri Reiniciáveis

Quando a rede de Petri for reiniciável e todas as barras forem potencialmente disparáveis, a rede é viva.

Isso pode ser verificada com a ajuda da máquina de senhas. Para o exemplo de rede de Petri da Fig. 3.1, a máquina de senhas é mostrado na Fig. 3.2.d.

Na seção anterior verificou-se que esta rede é reiniciável para a dada marcação inicial M0, portanto à partir de qualquer marcação M pertencente a {D(M0)}, existe um sequência de disparo que leva qualque Mj pertencente a {D(M0)}.

Para verificar a propriedade de vivacidade desta rede de Petri, basta verificar se todas as barras pertencente a RP são potencialmente disparáveis.

O cojunto de barras de RP da Fig. 3.1 são {b1, b2, b3, b4, b5, b6, b11, b12, b13, b14, b15, b16, b17}.

Na Fig. 3.2d, pode-se verificar que na marcação M0 a barra b1 é habilitada a disparar para atingir a marcação M1, e na marcação M3, a barra b2 é disparada para atingir a marcação M5.

Repetindo este procedimento, pode-se encontrar o conjunto de barras que são habilitadas a disparar pelo menos uma vez em

qualquer marcação da [D(M0)]. Neste exemplo, verifica-se que todas as barras que pertence a rede de Petri são disparáveis, portanto esta rede é viva.

b) Redes de Petri não Reiniciáveis.

Definição : Máquina de senha fortemente conexa

Uma máquina de senha é chamada fortemente conexa se existir, para duas marcações quaisquer i e j consideradas nessa ordem, um caminho com início em i e com término em j [9].

Uma máquina de senha qualquer S de X marcações pode ser divididas em várias sub-máquinas de senhas com X1, X2 ... , Xq marcações cada uma. As sub-máquinas de senhas S1, S2 ..., Sq de S geradas respectivamente pelos X1, X2 ..., Xq, que são fortemente conexas individualmente, são chamadas componentes fortemente conexas (CFC).

Seja por exemplo a máquina de senha mostrada na Fig. 3.12. Os sub-conjuntos fortemente conexos são três : X1 = [M1, M2, M3, M4 e M5] , X2 = [M7, M8] e X3 = [M6].

As sub-máquinas de senha geradas por esses subconjuntos são mostradas na Fig. 3.13.

Para a análise da vivacidade, são necessário encontrar os componentes fortemente conexos isolados (CFCI). Estes componentes apresentam uma característica de que à partir de um destes CFCI não existe nenhum caminho para acessar outro CFC.

O S2 da Fig. 3.12 é um exemplo de CFCI, pois este é formado por marcações M7 e M8 , e a partir destas marcações não existe nenhum caminho para atingir outra marcação diferente destas.

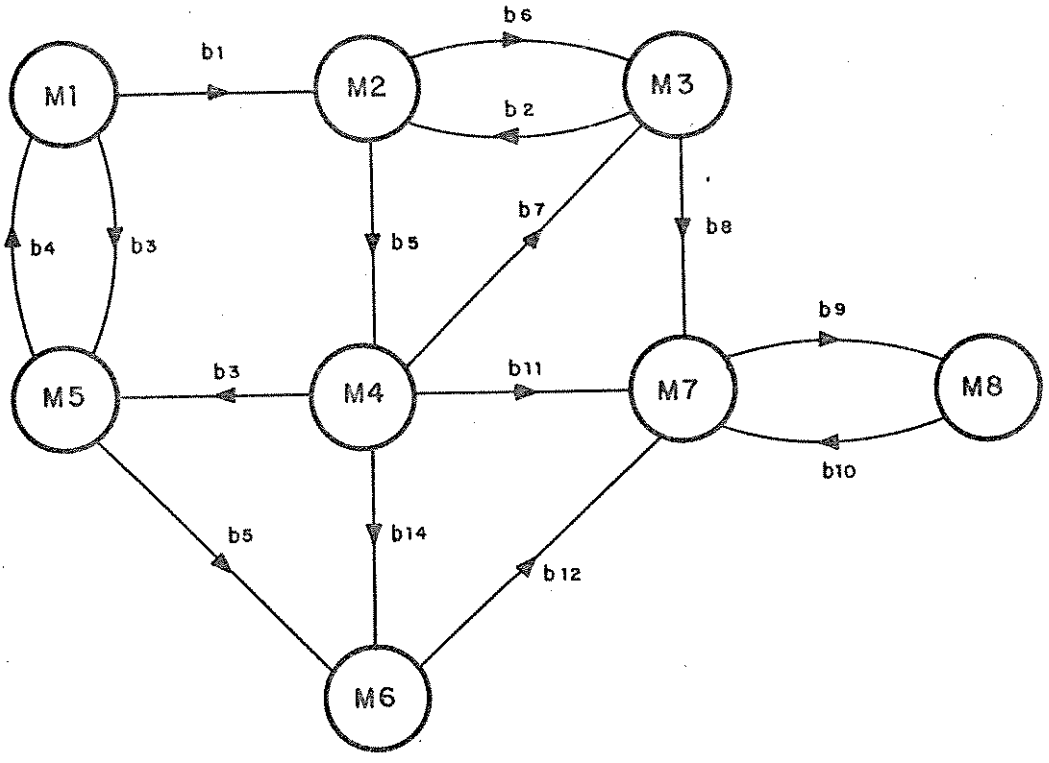


Fig. 3.12 - Máquina de senha

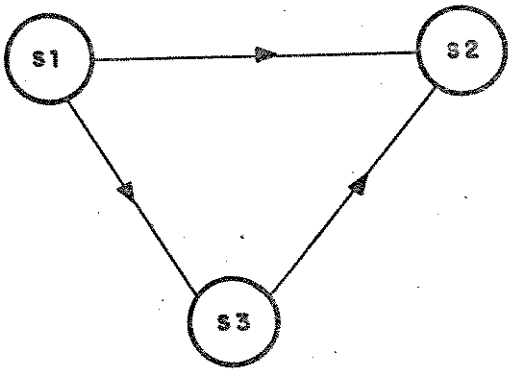


Fig. 3.13 - Componente Fortemente Conexas (CFC)

Uma rede de petri não reiniciável será viva se cada um dos CFCI contiver todas as barras da RP potencialmente disparáveis.

Prova : Sejam S = {S1, S2 ... Sq} o conjunto dos componente fortemente conexos e F = {F1, F2 ... Fp}, p <= q, um subconjunto de S, tal que os componentes de F contenham todas as transições

da rede. Portanto, as sub-máquinas de senhas de F são possíveis de disparar toda e qualquer transição da rede.

Além disso, como os outros componentes de S têm arcos que se dirigem aos componentes de F, podemos dizer que a partir de qualquer marcação é possível disparar uma transição qualquer. Portanto, a rede é viva.

No exemplo da Fig. 3.12, a rede é não viva pois no componente fortemente conexo isolado S2, são rotulados apenas as barras b9 e b10.

A Fig. 3.14 mostra o fluxograma do algorítmo implementado para verificar a propriedade de vivacidade de redes. Na figura pode-se verificar que antes de verificar esta propriedade deve verificar a propriedade de reiniciabilidade.

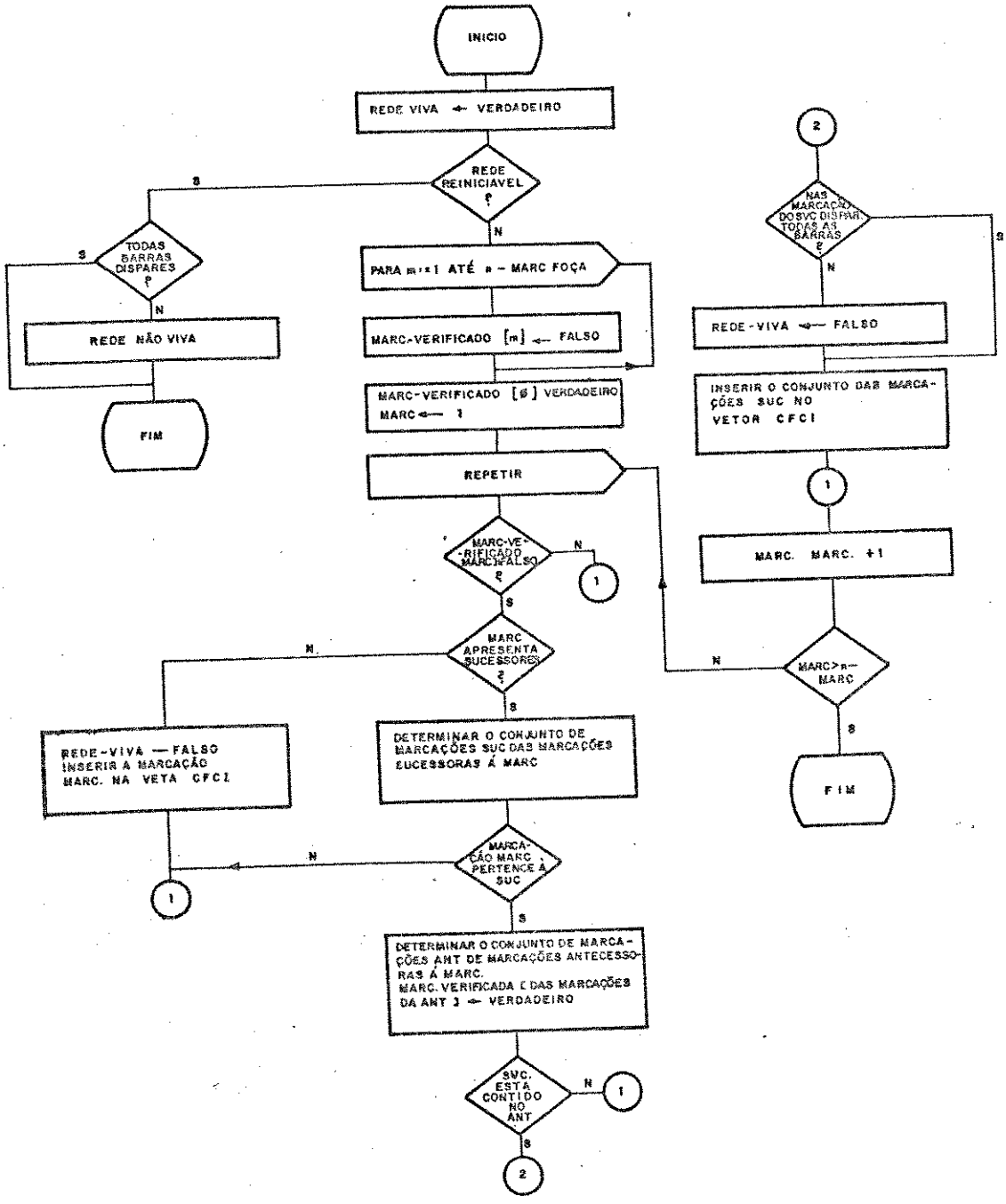


Fig.3.14 - Fluxograma do algoritmo para verificar a propriedade de vivacidade de RP.

3.4 - Estrutura do Programa Implementado no VAX (SIPRO-VAX)

3.4.1 - Concepção

O SIPRO-VAX é um pacote de software que analisa as propriedades clássicas da RP para auxiliar na validação de protocolos de comunicação. Ele aceita como dados de entrada, o protocolo descrito em diagrama SDL em uma forma codificada. Na saída, o programa apresenta as estruturas do protocolo em SDL na forma codificada e a de rede de Petri equivalente, além dos resultados da análise das propriedades clássicas da RP.

O programa permite, também entrar com os dados decodificados de uma rede de Petri qualquer para a sua análise.

SIPRO-VAX é constituído por quatro programas de computador, a saber : SDLVAX, DESCVAX, ANAVAX e DOCVAX.

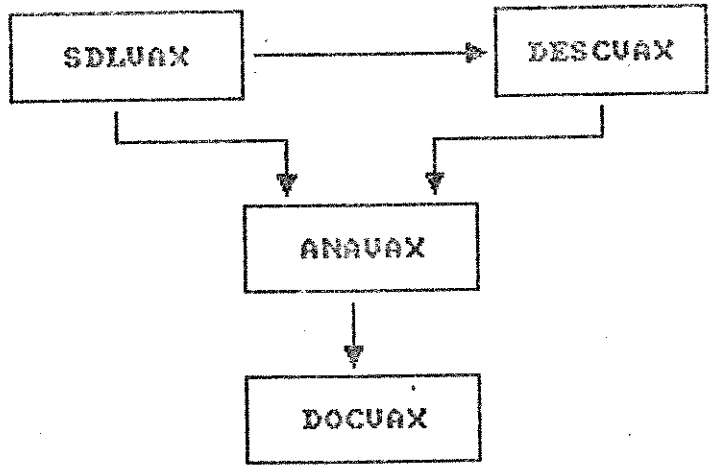


Fig. 3.15 - Diagrama de blocos mostrando o interfaceamento entre os programas do SIPRO-VAX.

A Fig.3.15 mostra as interações entre os quatro programas.

Estas interações são feitas por intermedio de arquivos de interfacemento criados pelos próprios programas, cabendo ao usuário definir os nomes destes arquivos.

O programa denominado SDLVAX possibilita de um modo interativo descrever, alterar e documentar um protocolo descrito em linguagem SDL, além de realizar a conversão do protocolo para a sua representação em RP.

O programa denominado DESCVAX possibilita de um modo interativo alterar, descrever e documentar uma ou mais rede de Petri. Neste programa é feita a análise de redução para diminuir o número de marcações que a RP pode atingir à partir de uma dada marcação inicial.

O programa ANAVAX gera a tabela de marcações, a máquina de senhas e analisa as propriedades clássicas de RP (limitabilidade, vivacidade e reiniciabilidade).

O programa denominado DOCVAX faz a documentação dos resultados da análise feito no ANAVAX, sendo que o usuário especifica o nível de detalhamento desejado.

3.4.2 - Programa de Computador : SDLVAX

Para o SDLVAX, o protocolo é representado em termos de Blocos Funcionais, processos, ramos e símbolos, definidos na seção 2.2.

A seguir, serão apresentadas as estruturas básicas das variáveis utilizada no programa de computador para armazenar as informações necessária para cada um destes termos. Em seguida são apresentados os principais cardápios de opções e a implementação do conversor.

a) Símbolo

Em diagrama SDL, o símbolo é identificado por nome, tipo e processo que pertence. Neste programa, um símbolo é identificado pelo seu código, nome e tipo.

Para armazenar estes dados, foi utilizado um vetor SIMB com a seguinte estrutura:

```
SIMB  :  array[1..max_simb] of tipo_simb,

tipo_simb  = record
               nome : string[40];
               tipo : (nulo, est, ee, ei, t, dec, se, si)
             end;
```

Este vetor tem dimensão max_simb (500), e cada posição corresponde a código do símbolo. Em cada posição são armazenados o nome do símbolo que pode ter até 40 caracteres e o seu tipo (que podem ser estado, entrada externa, entrada interna, tarefa, decisão, saida externa, saida interna ou nulo). Tipo nulo significa um símbolo não utilizdo n diagrama.

Como exemplo, considere os símbolos 1 e 12 da Fig.2.3. Nestes casos temos :

```
SIMB[1].nome = ativo_usuário
SIMB[1].tipo = est (estado)

SIMB[12].nome = reconhecimento de religação
SIMB[12].tipo = ee  (entrada externa)
```

b) Ramos

Para armazenar os dados dos ramos foi utilizado o vetor RAMO, que apresenta a seguinte estrutura :

```
RAMO : array[1..max_ramo] of tipo_ramo;

tipo_ramo = record
              num_simb : integer;
              set_simb : array[1..8] of integer;
            end;
```

Este vetor tem a dimensão max_ramo (200) e em cada posição que corresponde ao código do ramo são armazenados o número de símbolos que está contido neste ramo e os códigos dos respectivos símbolos.

Os códigos dos símbolos que pertencem ao ramo devem ser fornecidos sequencialmente. Isto é, o primeiro código deve ser de um símbolo estado onde inicia o ramo, seguido de um código de símbolo de sinal de entrada e por último um código de símbolo estado, onde termina o ramo.

Como exemplo considere o ramo 13 da Fig. 2.3. Para este ramo temos:

```
RAMO[13].num_simb     = 4
RAMO[13].set_simb[1]  = 23
RAMO[13].set_simb[2]  = 33
RAMO[13].set_simb[3]  = 34
RAMO[13].set_simb[4]  = 21
```

c) Processos

Para os processos, foi utilizado o vetor PROCESSO, que apresenta a seguinte estrutura :

```
PROCESSO  : array[1..max_processo] of tipo_processo;

tipo_processo  = record
                    num_ramo  : integer;
                    set_ramo  : array[1..max_ramo] of integer;
                    nome      : string[40];
                    estado_inicial : integer;
                 end;
```

Este vetor é de dimensão max_processo (10) e em cada posição são armazenados o numero de ramos que pertence a este processo, os códigos dos respectivos ramos, o nome do processo e um código do símbolo de estado que representa o estado inicial do processo. Este código do símbolo de estado inicial é utilizado para criar a

marcação inicial na rede de Petri equivalente.

Como exemplo, considere o processo 2 da Fig. 2.3. Para este processo temos :

```
Processo[2].nome   =  Lado Rede
Processo[2].num_ramo = 7
Processo[2].set_ramo[1..7] = [11, 12, 13, 14, 15, 16, 17]
Processo[2].estado_inicial   = 21
```

d) Blocos Funcionais

Para os blocos funcionais foi utilizado o vetor B_FUNC, que apresenta a seguinte estrutura:

```
B_FUNC   : array[1..max_bloco] of tipo_bloco,

tipo_bloco   = record
                 num_proc   : integer;
                 set_proc : array[1..max_proc] of integer;
               end;
```

O vetor B_FUNC tem dimensão max_bloco (10) e em cada posição só armazenados o número de processos, e os respectivos códigos.

Como exemplo considere o Bloco funcional 2 da Fig. 2.3. Neste caso temos :

```
BLOCO[2].num_proc = 1
BLOCO[2].set_proc[1] = 2
```

3.4.2.1 - Execução do Programa

O programa SDLVAX foi estruturado para interagir de um modo conversacional com o usuário através de cardápios de opções, informando o proximo procedimento a ser seguido.

Na Fig 3.16 é mostrado os principais cardápios que encontra durante a execução.

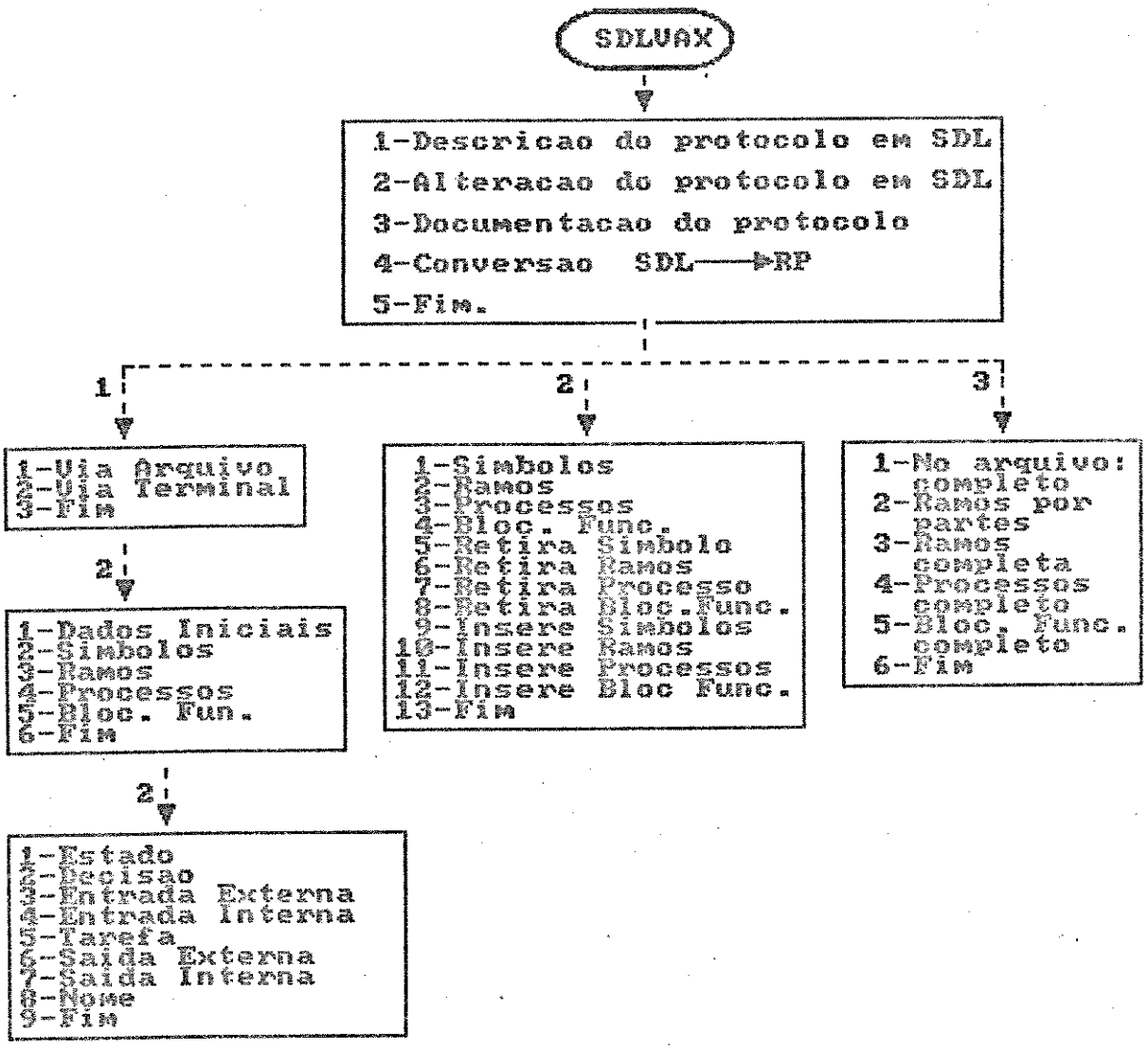


Fig. 3.16 - Cardápio de opções do SDLVAX em diagrama de blocos

Ao iniciar a execução, este programa permite a descrição do protocolo via arquivo (deve fornecer o nome do arquivo que contém a descrição do protocolo em SDL) ou via terminal.

Na descrição via terminal (o usuário deve ter em mão o diagrama do protocolo representado em SDL com símbolos, ramos,

processos e blocos funcionais enumeradas), encontra-se um sub-cardápio. Na opção Dados Iniciais devem ser fornecidos os códigos de símbolo, ramo, processo e bloco funcional que possuem enumerações mais alta.

Na descrição dos símbolos encontra-se um outro sub-cardápio, mostrado na Fig. 3.16. Neste subcardápio deve-se selecionar uma das opções e digitar os códigos dos símbolos correspondentes.

Para descrição dos ramos, processos e blocos funcionais, basta selecionar o número correspondente no cardápio e fornecer os respectivos dados necessários das variáveis SIMB, RAMO, PROCESSO E BLOCO discutidas anteriormente.

A opção alteração de protocolo, permite as modificações ou continuar descrição interrompida na opção de descrição.

O SDLVAX pode realizar a documentação do protocolo conforme as opções do sub-cardápio. A opção no arquivo permite documentação na impressora. A Fig. 3.17 mostra a documentação do protocolo da Fig. 2.3.

Na opção de conversão deve-se definir o nome do arquivo de interfaceamento para ser utilizado no programa DESCRI e/ou ANA.

À seguir, será apresentado o algoritmo implementado para realizar a conversão do protocolo descrito em SDL para RP.

```
        DOCUMENTACAO DO PROTOCOLO EM SDL

     DADOS INICIAIS

     NUMERO DE BLOCOS FUNCIONAIS = 2
     NUMERO DE PROCESSOS         = 2
     NUMERO DE RAMOS             = 17
     NUMERO DE SIMBOLOS          = 36

---> BLOCO FUNCIONAL : 1

--> PROCESSO : 1  NOME : Lado Usuario

      ESTADO INICIAL DO PROCESSO = SIMBOLO : 1

-> RAMO 1
CODIGO DO SIMBOLO - TIPO : NOME
   1 -       ESTADO       : Ativo-Usuario
   2 -  ENTRADA INTERNA   : Suspensao
   3 -    SAIDA EXTERNA   : Solicitacao de Suspensao
   4 -       ESTADO       : Pedido de Suspensao

-> RAMO 2
CODIGO DO SIMBOLO - TIPO : NOME
   4 -       ESTADO       : Pedido de Suspensao
   5 -  ENTRADA EXTERNA   : Reconhecimento de Suspensao
   6 -       TAREFA       : Libera Referencia de Chamada
   7 -       ESTADO       : Suspensao-Usuario

-> RAMO 3
CODIGO DO SIMBOLO - TIPO : NOME
   4 -       ESTADO       : Pedido de Suspensao
  13 -  ENTRADA EXTERNA   : Rejeicao de Suspensao
   1 -       ESTADO       : Ativo-Usuario

-> RAMO 4
CODIGO DO SIMBOLO - TIPO : NOME
   7 -       ESTADO       : Suspensao-Usuario
   8 -  ENTRADA INTERNA   : Religacao
   9 -       TAREFA       : Selecao de Nova Referencia de Chamada
  10 -    SAIDA EXTERNA   : Solicitacao de Religacao
  11 -      ESTADO        : Pedido de Religacao

-> RAMO 5
CODIGO DO SIMBOLO - TIPO : NOME
  11 -       ESTADO       : Pedido de Religacao
  12 -  ENTRADA EXTERNA   : Reconhecimento de Religacao
   1 -       ESTADO       : Ativo-Usuario

-> RAMO 6
CODIGO DO SIMBOLO - TIPO : NOME
  11 -       ESTADO       : Pedido de Religacao
  14 -  ENTRADA EXTERNA   : Rejeicao de Religacao
   7 -       ESTADO       : Suspensao-Usuario
```

```
---> BLOCO FUNCIONAL : 2

--> PROCESSO : 2  NOME : Lado Rede

      ESTADO INICIAL DO PROCESSO = SIMBOLO : 21

-> RAMO 11
CODIGO DO SIMBOLO - TIPO : NOME
  21 -       ESTADO       : Ativo-Rede
  22 -  ENTRADA EXTERNA   : Solicitacao de Suspensao
  23 -       ESTADO       : Pedido de Suspensao

-> RAMO 12
CODIGO DO SIMBOLO - TIPO : NOME
  23 -       ESTADO       : Pedido de Suspensao
  24 -  ENTRADA INTERNA   : Reconhecimento de Suspensao
  25 -    SAIDA EXTERNA   : Reconhecimento de Suspensao
  26 -       ESTADO       : Suspensao-Rede

-> RAMO 13
CODIGO DO SIMBOLO - TIPO : NOME
  23 -       ESTADO       : Pedido de Suspensao
  33 -  ENTRADA INTERNA   : Rejeicao de Suspensao
  34 -    SAIDA EXTERNA   : Rejeicao de Suspensao
  21 -       ESTADO       : Ativo-Rede

-> RAMO 14
CODIGO DO SIMBOLO - TIPO : NOME
  26 -       ESTADO       : Suspensao-Rede
  27 -  ENTRADA EXTERNA   : Solicitacao de Religacao
  28 -       ESTADO       : Pedido de Religacao

-> RAMO 15
CODIGO DO SIMBOLO - TIPO : NOME
  28 -       ESTADO       : Pedido de Religacao
  29 -  ENTRADA INTERNA   : Reconhecimento de Religacao
  30 -       DECISAO      : Canal B Original?
  31 -       TAREFA       : Libera Canal B Original
  32 -    SAIDA EXTERNA   : Reconhecimento de Religacao
  21 -       ESTADO       : Ativo-Rede

-> RAMO 16
CODIGO DO SIMBOLO - TIPO : NOME
  28 -       ESTADO       : Pedido de Religacao
  35 -  ENTRADA INTERNA   : Rejeicao de Religacao
  36 -    SAIDA EXTERNA   : Rejeicao de Religacao
  26 -       ESTADO       : Suspensao-Rede

-> RAMO 17
CODIGO DO SIMBOLO - TIPO : NOME
  28 -       ESTADO       : Pedido de Religacao
  29 -  ENTRADA INTERNA   : Reconhecimento de Religacao
  30 -       DECISAO      : Canal B Original?
  32 -    SAIDA EXTERNA   : Reconhecimento de Religacao
  21 -       ESTADO       : Ativo-Rede
```

Fig. 3.17 - Documentação do protocolo da Fig. 2.3

3.4.2.2 - Algoritmo de Conversão

Para obter uma rede de Petri equivalente de um protocolo aplicando-se as regras de conversão apresentadas na seção 3.2, deve ter um método para encontrar os conjunto de barras e de lugares necessários para representar o protocolo e conhecer os arcos orientados que interligam os lugares e as barras.

O conjunto de barras pode ser encontrada diretamente do protocolo em SDL, uma vez que cada ramo SDL corresponde a uma barra (Regra numero 3). Na implementação, os códigos dos ramos

foram mantidos para as barras correspondentes para facilitar na localização da barra no contexto do protocolo.

Para obter o conjunto de lugares de RP, foi utilizado uma tabela, denominada de tabela de lugares (T_L). Nesta tabela, cada linha representa o código do lugar e são armazenadas as informações referente a símbolo SDL que este lugar corresponde.

A tabela de lugares apresenta a seguinte estrutura:

```
T_L [I] = RECORD
            tipo : tipo_simbolo;
            nome : string[40];
            pro_bf : integer;
          END;
```

onde :

I -> Código do lugar da rede de Petri que corresponde a símbolo S.

nome -> nome do símbolo S.

tipo -> tipo do símbolo S.

pro_bf -> dependendo do tipo de S armazena o código (B) do bloco funcional ou código (P) do processo que S pertence.

Se o tipo de S é de estado ou sinal interno, armazena o código P de processo e caso de sinal externo o código B de bloco funcional.

O uso do T_L foi necessário para identificar os casos de vários símbolos SDL com nomes idênticos, mas tem lugares correspondentes distintas. Abaixo são apresentados dois destes casos.

1 - Os símbolos SDL de estado com nomes idênticos mas pertencentes aos processos diferentes.

2 - Os símbolos de sinais internos com nomes idênticos mas

pertencentes a blocos funcionais diferentes.

O método implementado para gerar a tabela T_L será explicado seguindo os procediemntos necessários no exemplo de protocolo fictício da Fig. 3.18.

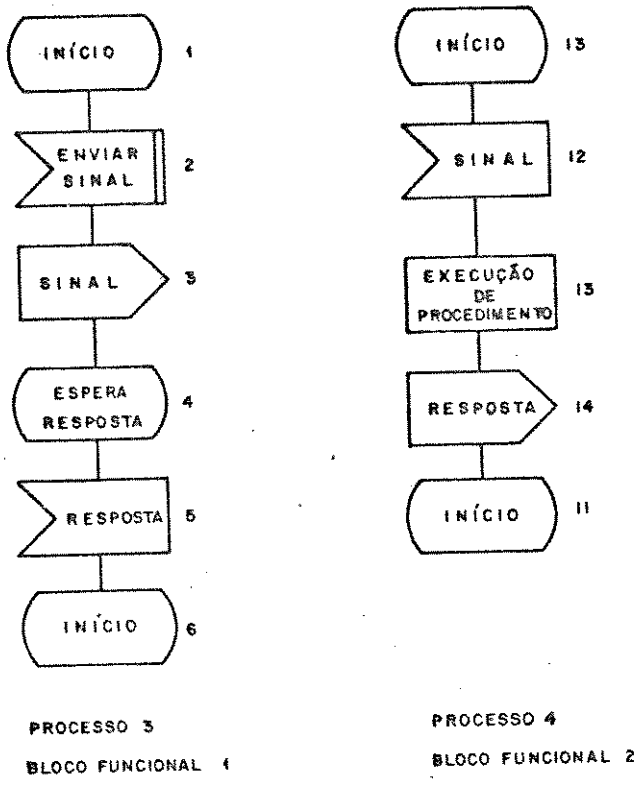


Fig. 3.18 - Exemplo de protocolo fictício.

Na montagem da tabela é feita uma análise de cada símbolo para verificar se terá um lugar correspondente na RP. Esta análise é inicializada nos símbolos pertencente a bloco funcional de menor código e segue em ordem crescente deste código.

Para cada símbolo em análise, é realizado uma consulta na T_L para verificar se já existe o seu lugar correspondente. Para fazer esta consulta, o tamanho da tabela (número de linhas) é controlado por contador de lugares criados (CONT_LUG). Este

contador é inicializado por valor nulo para indicar que a tabela está vazia.

A seguir são apresentados os procedimentos seguidos na montagem da T_L para cada símbolo do exemplo de protocolo acima.

Símbolo 1 (S1) - Tipo de estado e o seu nome é Início.

Na consulta da T_L, não é encontrado o lugar correspondente, pois a tabela está vazia. Portanto a variável CONT_LUG é incrementado e nesta linha são armazenadas seguintes dados : tipo = estado; nome = inicio; e PRO_BF = 3 (código do processo).

Símbolo 2 (S2) - Tipo de entrada interna com nome Enviar Sinal.

Na consulta á tabela não encontra o lugar correspondente, pois na tabela existe apenas o lugar correspondente ao S1.

Em seguida é verificado se existe um símbolo de saída externa com o mesmo nome nos outros processos pertencente ao mesmo bloco funcional. Como no BF1 contém apenas um processo, entende-se que o símbolo S2 não apresenta o lugar correspondente em RP.

Símbolo 3 (S3) - tipo de saída externa e com nome Sinal.

Como não existe o lugar correspondente na T_L, é feito uma verificação nos outros blocos funcionais se contém o símbolo de tipo entrada externa com nome igual do S3. Encontrado o símbolo S12 no processo 4, é incrementado o contador (CONT_LUG = 2) e nesta linha são armazenada os seguintes dados: tipo = saída externa, nome = Enviar Sinal, e PRO_BF = 1 (código do bloco funcional).

Símbolo 4 (S4) - tipo de estado com nome Espera Resposta.

Como não existe um lugar correspondente na T_L, o contador é incrementado e nesta linha é armazenado os seguintes dados: tipo = estado; nome = Espera Resposta; e PRO_BF = 3.

Símbolo 5 (S5) - Tipo de entrada externa com nome Resposta.

Mesmo não existindo um correspondente na tabela, mas como o processo 4 do BF2 apresenta o símbolo correspondente representado por S14, o contador é incrementado (CONT_LUG = 4) e nesta linha são armazenadas os seguintes dados: tipo = entrada externa; nome = Resposta; e PRO_BF = 1.

Símbolo 6 (S6) - Tipo de estado e nome é Início.

Na consulta à tabela encontra um lugar correspondente represento por I1, portanto não é criado um novo lugar.

Símbolo 11 (S11) - Tipo de estado com nome Início.

Na consulta à tabela, não é encontrado nenhum lugar correspondente, pois, neste caso o PRO_BF é igual 4. Portanto, o contador é incrementado e nesta linha são armazenados os dados deste símbolo.

Simbolo 12 (S12) - Tipo Entrada Externa com nome de Sinal.

Não será criado um lugar, pois é encontrado o lugar correspondente (I2) durante consulta à tabela.

Símbolo 13 (S13) - Tipo tarefa.

Nos símbolos deste tipo não é executado nenhum procedimento.

Símbolo 14 (S14) - Tipo de saída externa com nome Resposta.

Este símbolo já apresenta um lugar correspondente na T_L representado por I4.

Conhecido o método para encontrar os conjunto de barras e de lugares, rede de Petri equivalente pode ser encontrada aplicando regra número 3 da seção 3.2 em cada ramo do protocolo.

Finalmente, para obter a marcação inicial é procurar, na T_L, os lugares que corresponde a um símbolo de estado representando o estado inicial de cada processo.

A marcação inicial é armazenada em um vetor com nome MARC_INICIAL. Este vetor é do tipo BLOCO (apresentado na seção 3.3.2), e em cada posição são armazenadas o código do lugar que possui a senha na MO e a quantidade de senha deste lugar. A marcação inicial da rede de Petri, que representa um protocolo, possui apenas uma senha por lugar.

3.4.3 - Programa de Computador : DESCVAX

O programa DESCVAX permite a descrição, alteração, documentação e redução de lugares de uma rede de Petri. Para realizar estas operações, o programa trabalha com as variáveis M_INC, M_REV, MARC_INICIAL definidas anteriormente e dois vetores, onde em cada posição do vetor pode ser armazenado um nome de barra ou de lugar.

Este programa foi estruturado em forma de cardápios. Assim, durante a execução encontram-se os cardápios mostrados na Fig. 3.19.

Tendo em vista que a maioria dos exemplos de rede de Petri apresentam os pesos dos arcos de entrada e de saída das barras igual a 1, resolveu-se fixar os peso como sendo 1, por "default". Assim, na descrição via terminal, pergunta-se antes ao usuário se deseja seguir o "default" ou não. Em seguida, o usuário deve

descrever as barras digitando o código da barra e os códigos dos respectivos lugares de entrada e de saída, assim como os códigos dos lugares que possui senhas de marcação inicial e o número de senhas. O usuário pode ainda optar em atribuir os nomes das barras e dos lugares. Uma vez realizada a descrição via terminal, cabe ao usuário definir ou não o sua armazenagem num arquivo de interfaceamento.

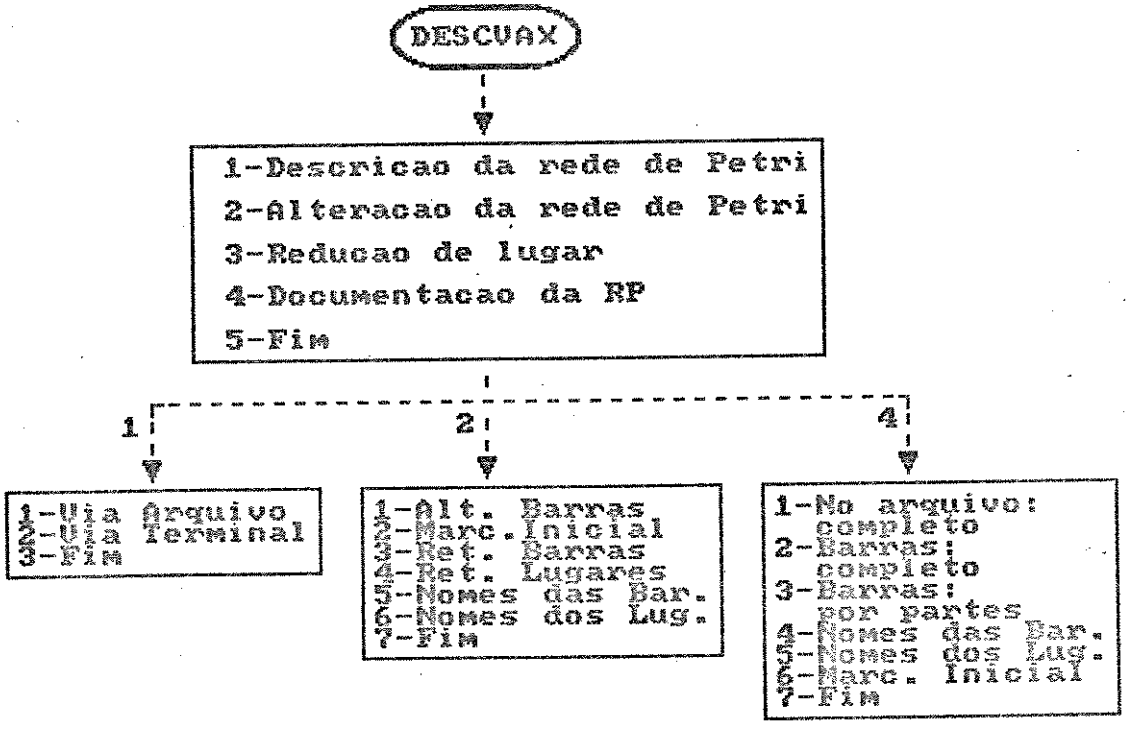


Fig. 3.19 - Cardápio de opções do DESCVAX

Após a descrição da rede de Petri, pode-se realizar várias alterações como : criar ou reestruturar as barras, alterar a marcação inicial, retirar barras ou lugares, assim como alterar nomes das barras e dos lugares.

Para a redução de lugares, cabe ao usuário definir ou não a

sua armazenagem em um arquivo de interfacemento. A redução de lugares é feita conforme o algorítmo apresentado na seção 3.3.4. Para cada lugar retirado da RP, o seu código é apresentado na tela.

A documentação da rede de Petri pode ser feita em um arquivo (opção 1 do sub-cardápio) ou na tela com o detalhamento requerido, escolhendo as opções entre 2 a 6 do sub-cardápio.

3.4.4 - Programa de Computador : ANAVAX

O programa ANAVAX monta a tabela de marcação (Tab_marc), a maquina de senha (Maq_Senha) e verifica as propriedades de rede de Petri.

Ao iniciar a execução, o usuário deve fornecer o nome do arquivo de interfaceamento criado por DESCVAX ou SDLVAX, e do arquivo de interfaceamento para armazenar o resultado da análise.

Em seguida, deve selecionar entre a montagem das tabelas completa ou parcial.

A opção da análise parcial foi implementada no programa porque o tempo de execução deste programa cresce muito com o número de marcações (estados) a ser calculado. Esta opção permite, ao usuário, a verificação parcial das marcações calculadas.

Ao verificar a ocorrencia da primeira marcação superior (rede não limitada) a execução do programa é interrompida e o usuário deve optar em continuar a análise ou o termino da execução.

Após a montagem completa de Tab_Marc e Maq_Senha, passam a verificar as propriedades de reiniciabilidade e vivacidade

seguindo os algorítmos apresentados na seção 3.3.6 e 3.3.7.

3.4.5 - Programa de Computador : DOCVAX

O programa DOCVAX é o que realiza a documentação da análise da rede de Petri. Sua utilização depende da prévia execução do programa ANAVAX, sendo que este gera o arquivo de interfaceamento necessário para a documentação.

Este programa, como os anteriores, é estruturado em forma de cardápios. A Fig 3.20 mostra o cardápio encontrado durante a execução.

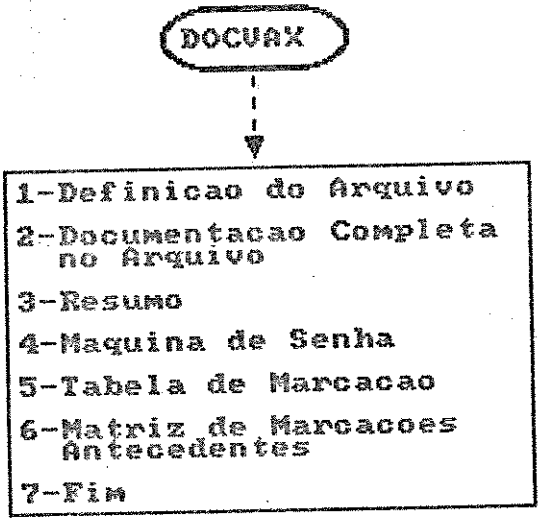


Fig 3.20 - Cardápio do DOCVAX.

A opção 1 deve ser escolhido em primeiro lugar, para definir o nome do arquivo de interfaceamento. Na opção 2 deve fornecer o nome de um arquivo para posteriormente ser documentado em uma impressora. Os ítens 3, 4 e 5 referem-se à documentação via terminal de video. A opção 3 fornece apenas os resultados dos testes de vivacidade, limitabilidade e reiniciabilidade da rede

de Petri.

Na opção 6, a documentação é feita em um arquivo, cabendo ao usuário definir o seu nome. Nesta opção, o usuário deve digitar um código de uma marcação (Mi), e no arquivo aparece a documentação de todas marcações que apresentam uma sequência de disparo para atingir à Mi.

3.5 - Conclusão

Neste capítulo, foram apresentadas, inicialmente, as regras que foram desenvolvidas para conversão dos protocolos especificadas em linguagem SDL para a rede de Petri. Em seguida mostrou-se que as matrizes utilizadas nas análise são esparsas, portanto é vantajoso o uso de alocaçao dinâmica.

Apresentou-se o pacote de "software" denominado de SIPRO-VAX e as estruturas de cada um dos programas. Foram, também, discutidos em detalhes os principais algorítimos implementados.

CAPÍTULO 4

EXEMPLO DE VALIDAÇÃO DE PROTOCOLO

4.1 - Introdução

Neste capítulo são apresentadas inicialmente a arquitetura e a estrutura de sinalização entre os processadores da central telefônica Trópico-R [11,12]. Em seguida, apresentado o protocolo de comunicação entre os processadores da Trópico-R. É, também, discutida uma metodologia de validação dos protocolos que será utilizada para a validação do protocolo citado.

4.2 - Arquitetura da Trópico-R

Em geral, uma central telefônica pode ser esquematizada como mostra a Fig. 4.1.

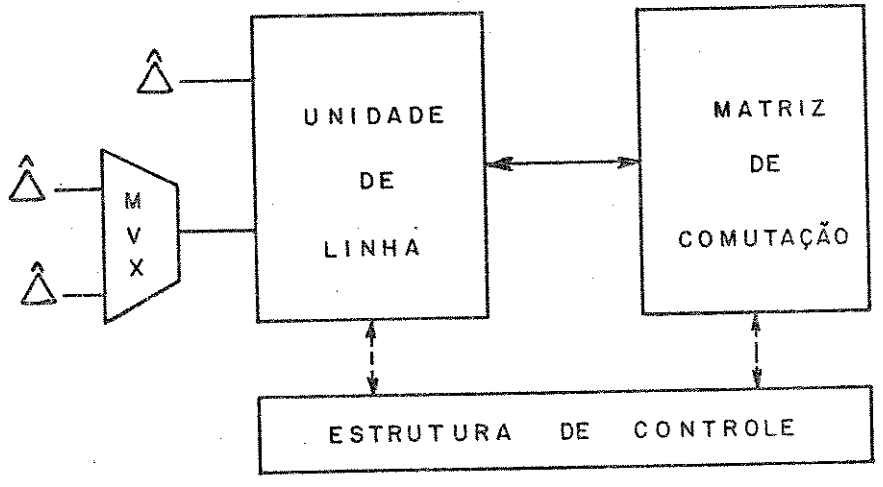


Fig. 4.1 - Esquema de uma Central Telefônica.

A unidade de linha é capaz de detectar os eventos telefônicos provocados pelos terminais de assinantes (ou tronco, ou linhas multiplexadas, etc.).

A unidade de linha faz a detecção dos eventos telefônicos através de uma varredura periódica dos terminais. As informações obtidas pela unidades de linha são encaminhadas à estrutura de controle para as suas análises.

A matriz de comutação estabelece um caminho entre dois terminais para permitir a sua conversação. A escolha de um caminho livre entre os terminais é feita pela estrutura de controle.

Assim a estrutura de controle tem como principais funções:

1 - O estabelecimento e supervisão de chamadas, que compreende por exemplo : supervisão de linhas de assinantes, recepção de dígitos, taxação, etc.

2 - Supervisão de alarmes para detectar os casos de falhas de equipamentos.

3 - Manutenção. São as tarefas de supervisão de elementos externas à central, como a linha de assinante.

4 - Operação. Permite ao operador mudar o estado operacional de terminais, modificação de tabelas de encaminhamento, etc.

Para a escolha da arquitetura da estrutura de controle deve-se levar em consideração:

1 - A estrutura física : Indica se a arquitetura é mono ou multi processador, além da hierarquia entre processadores.

2 - A estrutura funcional : Dada a estrutura física, indica como é a repartição das funções entre os componentes da estrutura física.

Quanto a estrutura física pode ter as seguintes classificações:

1 - Arquitetura centralizada: Um único processador gerencia todas

as tarefas referente à central telefônica.

2 - Arquitetura descentralizada: O processador central delega parte de suas tarefas a processadores regionais com funções específicas.

3 - Arquitetura distribuída: Vários processadores independentes que dividem as diversas funções realizadas durante o estabelecimento de uma chamada. Essa solução apresenta a vantagem de modularidade, podendo crescer com a capacidade instalada no sistema.

As primeiras centrais de comutação telefônica utilizaram arquiteturas centralizadas. Atualmente, com a grande disseminação dos micro e mini-computadores, estão sendo utilizadas as arquiteturas distribuídas. As arquiteturas distribuídas permitem uma alta confiabilidade contra falhas no sistema de comutação. Pois, ao contrário da arquitetura centralizada, uma falha em uma parte do circuito, não prejudica totalmente à central, podendo outras partes funcionarem sem problemas.

A central local de pequeno porte (Trópico-R) desenvolvida pelo Centro de Pesquisa e Desenvolvimento da Telebrás utiliza uma arquitetura distribuída.

A seguir, é apresentada a arquitetura da Trópico-R.

A Trópico-R é uma central telefônica digital. Insere-se dentro da família de sistemas de comutação denominada de CPA-T (temporais com controle por programas armazenadas).

A Trópico-R foi desenvolvido usando-se um sistema de comutação com controle completamente distribuído, implementado por microprocessadores.

A família de centrais Trópico é implementada à partir de

módulos construtivos básicos. Estes módulos são definidos em termos de funções e tráfego.

Para implementação da Trópico-R, apenas três tipos de módulos são requeridos:

- Módulo de Terminais (MT).
- Módulo de Operação e Manutenção (MO).
- Módulo de Comutação (MC).

Cada módulo é constituído por sub-estrutura denominadas sub-módulo, cada qual controlado por um processador. Um MT ou MO pode conter até quatro sub-módulos e um MC até cinco sub-módulos.

A configuração básica de um módulo é mostrada na Fig. 4.2.

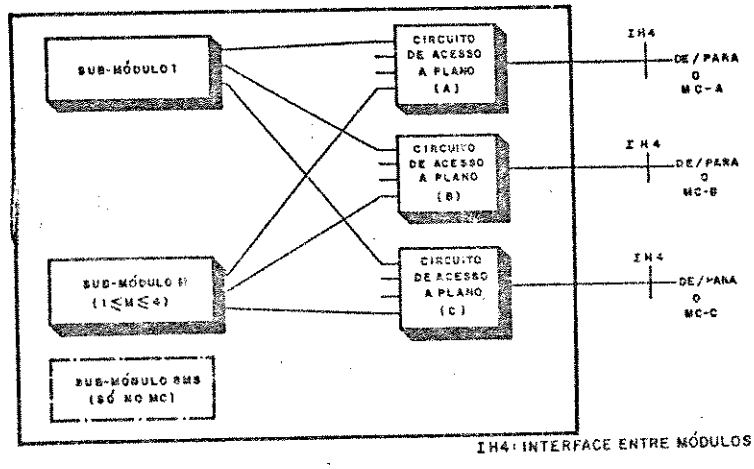


Fig.4.2 - Configuraçao Básica de um Módulo.

O módulo terminal faz o relacionamento da central com o meio externo, no que se refere as funções telefônicas. O MT prove o controle e os circuitos necessários para compatibilizar, informações trocadas com o meio externo, com a estrutura interna da central.

Com uma mesma estrutura básica, o MT pode ser configurado

para permitir a conexão de diferentes tipos de terminais à central.

Dentre os diversos tipos de sub-módulos de terminais que podem ser acomodados em um MT destacam-se os seguintes:

- SUB-MÓDULO DE ASSINANTES LOCAIS (SMT-A).
- SUB-MÓDULO DE TRONCOS ANALÓGICO (SMT-J)
- SUB-MÓDULO DE TRONCOS DIGITAL (SMT-D)
- SUB-MÓDULO DE ASSINANTE DISTANTE (SMT-T)
- SUB-MÓDULO DE SINALIZAÇÃO MFC(SMT-F)

O módulo de operação e manutenção é a parte da central que se relaciona com o meio externo para fins de operação e manutenção da central Trópico-R e supervisão e gerência de rede. O MO prove também a adequação dos sinais de entrada e de saída dos periféricos com a estrutura interna da central.

Um MO pode ser configurado de várias formas, dependendo dos periféricos utilizados e das funções de supervisão e gerência implementadas. Os principais sub-módulos são:

- SUB-MÓDULO DE PERIFÉRICOS (SMP)
- SUB-MÓDULO DE ROBO DE TESTE (SMT-T)
- SUB-MÓDULO AUXILIAR

O módulo de comutação é a parte principal da central Trópico-R, onde são realizadas a comutação de voz, comutação das mensagens de sinalização entre os processadores da central e geração e distribuição dos sinais de sincronismo.

As partes de hardware do MC que realizam a tarefa de comutação de voz, das mensagens de sinalização e geração e distribuição de sincronismo são denominadas respectivamente de plano de voz, plano de sinalização e plano de sincronismo. A união destes três planos resulta no que se denomina simplesmente Plano.

A Trópico-R apresenta três planos denominados de Plano A, B e C. Isto é, a Trópico-R contém três modulos de comutação (A, B e C), cada um relativo a um plano, trabalhando em partição de cargas.

Os principais sub-módulos contidos em um MC são:

- SUB-MÓDULO DE COMUTAÇÃO E SINCRONISMO (SMT-X)
- SUB-MÓDULO DE SINALIZAÇÃO (SMA-S)

Na Fig. 4.3 é apresentada a estrutura Hardware de uma central Trópico-R, com principais módulos interligados.

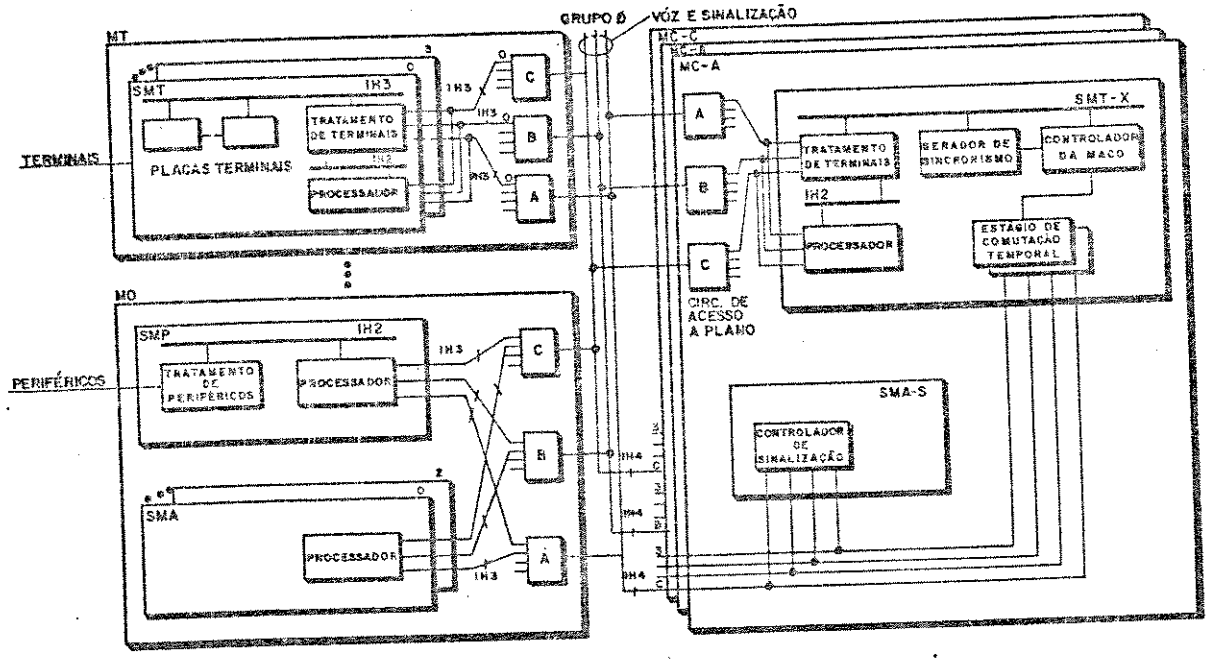


Fig. 4.3 - Interconexões entre os Módulos da Trópico-R.

Na Fig. 4.4 é mostrada a estrutura de uma central Trópico-R, com seus módulos construtivos básicos.

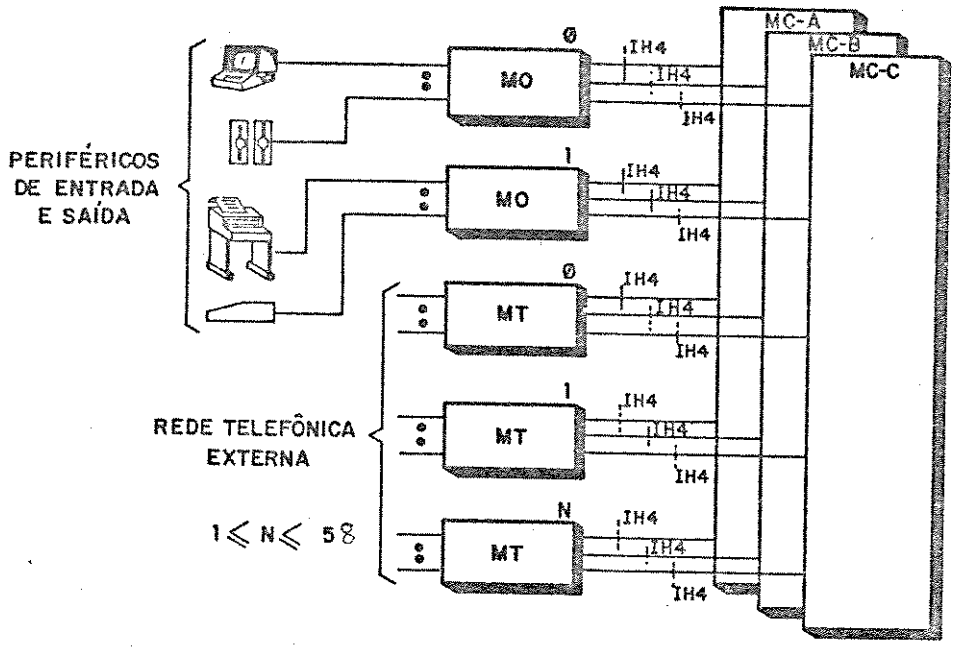


Fig. 4.4 - Arquitetura Básica da Trópico-R.

4.3 - Estrutura de Sinalização entre processadores da Trópico-R

Como foi visto na seção anterior, a Tropico-R é constituída por um grande número de processadores de forma distribuída, onde cada processador realiza funções de acordo com a necessidade.

Para esta arquitetura da central é necessário um "software" que apresenta como principais características o multi-processamento, o multi-programa de estrutura modular com interfaces padronizados e o controle distribuído.

A estrutura de "software" de controle da central baseia-se no conceito de máquina de estado, onde os recursos a serem controlados (terminais telefônicas, troncos, etc.) assumem vários estados durante uma chamada. Cada máquina de estado caracteriza um processo, e o conjunto de todos os processos associados a um tipo de recurso é chamado de bloco de implementação (BI). Estes

BI's são distribuídos convenientemente em cada processador, dependendo da sua função.

Para que esta estrutura de controle possa ser implementada, há em cada processador um sistema operacional denominado núcleo "software". Este núcleo prove os mecanismos de controle que são realizados através das funções como a comunicação entre processos, temporização, etc.

A comunicação entre os processos dos BI's "software" é feita exclusivamente através de sinais. Estes sinais consiste de três campos distintos : destino (processador, BI, processo, etc.), dados contendo as informaçoes para o processo, e origem de sinal (processador, BI, etc).

A seguir, é apresentada a estrutura física de sinalização entre os processadores para a realização da transmissão do sinal de um processo para outro processo residente em processador diferente.

Na estrutura física da via de sinalização da Trópico-R destacam-se os seguintes elementos:

CTS - Controlador de terminais e sinalização.
IAP - Interface de acesso à plano.
CVS - Controlador de vias de sinalização.

Um conjunto de quatro CTS's está ligado a um conjunto de 3 IAP's (A/B/C); cada IAP alocada à um plano de comutação. O conjunto de quatro CTS's e 3 IAP's é parte constituinte de um módulo terminal(MT).

Cada CTS está ligado a uma dada IAP através de 4 vias de sinalização intramodular: 1 via de transmissão, 1 via de solicitação, 1 via de recepção e 1 via de habilitação. De um dado

CTS saem 12 vias de sinalização, uma vez que o mesmo está ligado à três IAP's.

Cada IAP está ligado a CVS do seu plano através de 4 vias de sinalização intermodular: 1 via de transmissão, 1 via de solicitação, 1 via de recepção e 1 via de habilitação.

Na Fig 4.5 são mostradas, simplificadamente, as vias de comunicação entre um CTS e uma IAP, e entre ela e o seu correspondente CVS. Por essas vias percorrem sinais de sinalização.

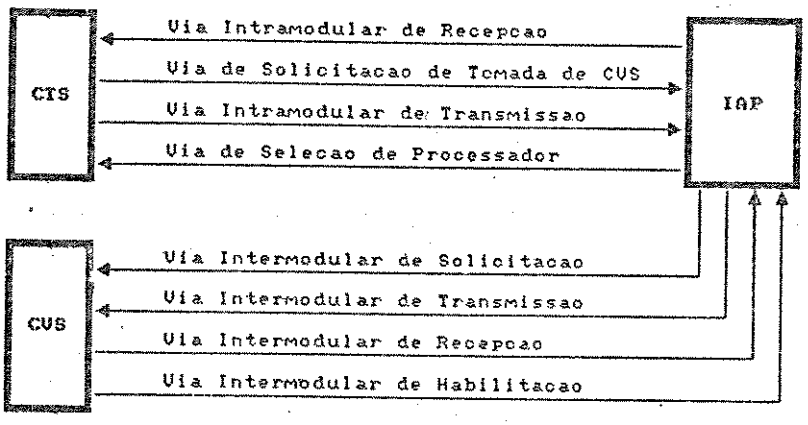


Fig. 4.5 - Vias Intra e Intermodulares de Sinalização.

CTS - Controlador de terminais e sinalização.

Os CTSs são processadores que realizam as tarefas de transmissão e recepção de mensagens trocadas entre os elementos de controle. Portanto a interação (envio e recepção de sinal) entre processos pertencentes a processadores diferentes é representado pela comunicação entre dois CTSs, um funcionando como transmissor e outro como receptor.

IAP - Interface de acesso a plano.

A IAP é um circuito passivo, que apresenta função de acoplar o processador a uma das vias intermodulares.

Os modos de operação da IAP estão entre o pedido da via de transmissão, transmissão, recepção e liberação (desconexão de processador com vias de sinalização). Estes modos de operação são estabelecidos pelo CVS do Plano correspondente.

CVS - Controlador de vias de sinalização.

Os CVSs são responsáveis pela alocação das vias de sinalização aos CTSs. O conjunto de 16 vias de sinalização é disputado por até 256 processadores; esta disputa é gerenciada pelos 3 controladores de vias de sinalização.

Na Fig. 4.6 é mostrado o caminho percorrido pelos sinais de sinalização na comunicação entre processadores.

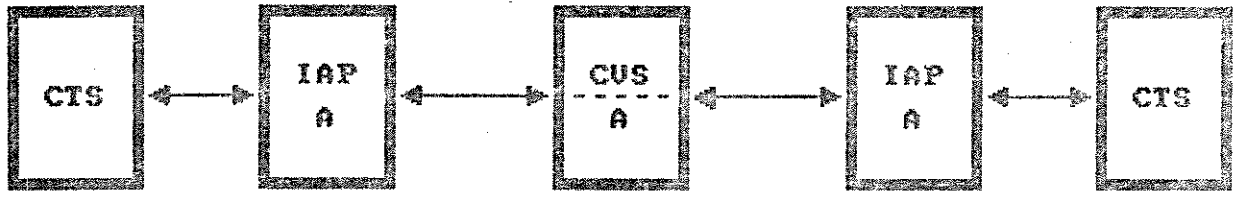


Fig. 4.6 - Caminho Percorrido pelos Sinais de Sinalização.

4.4 - Protocolo de Comunicação entre os Processadores

O núcleo "software", residente nos processadores, é interrompido a cada 4 mseg para tratamento de sinais. Existindo um sinal para ser tratado, é acionado um processo (primitiva TRANEX) responsável pela transmissão de sinal para outro processo.

Para efetuar-se a comunicação entre os processos pertencentes a processadores diferentes é necessário estabelecer a comunicação entre os processadores envolvidos. Esta comunicação é realizada através de um barramento que liga os diversos processadores, e de um protocolo bem definido que permite o acesso destes processadores ao barramento.

A recepção de um sinal pode ocorrer em qualquer instante, sendo necessário que o processador receptor encontre-se com a interrupção habilitada. Durante a recepção, o processador permanece em estado de interrupção desabilitada, e no caso de um outro processador tentar enviar um sinal, o "hardware" do processador responderá "NOK" indicando que o processador de destino está recebendo um sinal.

A transmissão de um sinal divide-se em quatro etapas: solicitação de vias, solicitação do processador de destino, transmissão de sinal e liberação de vias.

A seguir, são apresentadas as fases de transmissão de sinal. Neste protocolo são omitidos todos os temporizadores.

4.4.1 - Solicitação de Vias

Tendo acionado o processo responsável pela transmissão de sinal (TRANEX), o CTS de origem verifica quais os planos de

sinalização que estão desbloqueados (consultando uma tabela de configuração) e solicita vias de transmissão nestes planos.

Um plano pode ser bloqueado pelo operador ou em caso de falhas persistente detectadas pelo "software".

Quando a IAP recebe palavra de habilitação de varredura do CVS, o pedido de solicitação de vias do processador de origem é enviado para o CVS através da via de solicitação.

O "hardware" do CVS identifica o sinal de solicitação e, no caso em que o CTS de origem não esteja bloqueado na tabela de inibição de varredura, gera um sinal "hardware" que interrompe o processador de CVS. Este procedimento é necessário para evitar que um processador em falha permaneça solicitando indefinidamente.

O CVS identifica IAP e CTS de origem, e envia a palavra de habilitação de transmissão através da via de habilitação.

No caso de mais de um plano atender a solicitação de vias, o CTS de origem escolherá um dos planos utilizando o critério implementado. Após a escolha do plano, as solicitações de vias dos demais planos são retiradas, e o "hardware" de origem é programado para operar no modo de transmissão que envia o primeiro "byte", contendo o número do processador de origem na central, através da via de transmissão intramodular até a IAP e pela via de transmissão intermodular da IAP para o CVS.

Após o envio da habilitação de transmissão, o CVS fica verificando se chegou um "byte" ou se o processador de origem cancelou a solicitação de vias. Caso ocorra um cancelamento de solicitação de vias, o CVS envia a palavra de habilitação de liberação para a IAP de origem e reinicia a varredura das IAP's a

partir da IAP seguinte.

O processador de origem pode cancelar a solicitação de vias se ocorrer uma solicitação para recepção antes do segundo "byte" ter sido enviado, ou houve uma escolha de outro plano para a transmissão.

Após o recebimento do primeiro "byte", o CVS verifica se o número coincide com o endereço do processador que foi habilitado. Em caso positivo, o "hardware" do CVS é acionado par enviar "OK" através da via de recepção intermodular para a CTS de origem. Caso contrário, é enviada uma resposta "NOK".

O processador de origem após o envio do primeiro "byte", fica verificando a chegada de resposta referente ao primeiro "byte". Se chegar a resposta "NOK", significa que o CVS não está funcionando corretamente, uma indicação de falha será fornecido ao BPR ("software" que pertence a BI, que contém a função de manter a configuração dos processadores da central), e inicia a solicitação de vias nos outros planos. Por outro lado, se chegar a resposta "OK", é enviado o segundo "byte" contendo o número do processador de destino.

O CVS identifica a chegada do segundo "byte", e verifica se o endereço do processador de destino coincide com o seu próprio endereço. No caso de coincidência, o CVS responde "OK" e passa a receber o sinal. Caso o processador de destino seja um outro CVS, (caso de manutenção) envia uma resposta "NOK" para o processador de origem. NO caso em que o endereço de destino é um CTS, o CVS interliga a via de transmissão e recepção intermodular e envia a palavra de habilitação de recepção para IAP/CTS de destino.

4.4.2- Solicitação do processador de destino

O CVS envia a palavra de habilitação de recepção para o CTS de destino através da via de habilitação intermodular. A IAP ao receber esta palavra, ativa um sinal para a interrupção do processador do CTS que está sendo selecionado para a recepção.

A partir do instante que o processador atende a interrupção, o "software" do núcleo salva o contexto (registradores, PC, etc.). E o processador de destino identifica o plano que o solicitou para recepção, e bloqueia os demais planos para recepção, e envia a resposta "OK" para o processador de origem.

O CTS de origem que estava verificando a chegada da resposta relativo ao segundo "byte", pode receber o "NOK" indicando que o destino é um outro CVS ou que o processador de destino está recebendo um sinal através do outro plano. Nestes casos, retira a solicitação de vias, e espera a palavra de habilitação de liberação e inicia a solicitação de vias novamente. Caso receber resposta "OK", o CTS envia o segundo "byte" para o processador de destino iniciando fase de transmissão de sinal.

4.4.3- Trasmissão do sinal

A fase da transmissão inicia-se no instante que o processador de origem envia o segundo "byte" (com o mesmo conteúdo do byte que foi enviado para CVS) para o processador de destino após ter recebido o "OK" indicando que o processador de destino está programado para o modo de recepção. O processador de destino recebe o segundo "byte", verifica e envia "OK" caso o endereço esteja correto.

O processador de origem recebendo "OK" envia um outro "byte"

para o processador de destino.

O processador de destino recebe este outro "byte", e envia a resposta "OK" para o processador de origem.

Este procedimento é repetido até que a transmissão do "byte" de "check-sum" é enviado.

O processador de destino recebe o "check-sum", e verifica se o sinal é consistente, ou seja, "check-sum" correto. Caso positivo, uma resposta "OK" é enviada, caso contrário é enviada a resposta "NOK" para o processador de origem.

O processador de origem identifica a resposta relativa ao "byte" de "check-sum". A resposta "OK", significa que o sinal foi transmitido corretamente. Caso receba "NOK", o processador de origem transmitirá novamente a mensagem para o processador de destino.

4.4.4 - Liberação de vias

O processador de origem ao receber a resposta relativa "check-sum", retira a solicitação de vias. A IAP identifica a retirada de solicitção de vias e envia através da via de solicitação intermodular, o novo conteúdo para o CVS.

O processador de CVS ao identificar que a solicitação de vias foi retirada, envia a palavra de habilitação de liberação para o IAP do processador de destino e em seguida, carrega o contador de IAP's com o endereço da IAP de origem, envia a palavra de habilitação de liberação para IAP de origem e reinicia a varredura a partir da IAP seguinte.

4.5 - Metodologia de Análise para a Validação de Protocolo

A metodologia de análise para a validação de um protocolo utilizando o SIPRO-VAX consiste de seguintes etapas.

a) Converte-se o protocolo especificado em linguagem natural para a linguagem de projeto conhecida de SDL.

b) Aplica-se as regras de conversão apresentadas na seção 3.2 no protocolo especificado em SDL para obter a sua representação em Rede de Petri. Esta etapa é feito utilizando o programa SDLVAX.

c) Verifica-se as propriedades clássica da RP, com auxílio de ANAVAX.

d) Com o auxilio do programa DOCUMVAX, pode verificar se ocorreu alguma imperfeição na rede (ou seja, não verificação de alguma propriedade). Caso ocorrer uma imperfeição, verifica-se qual o problema existente, corrige-se o protocolo original e reinicia a etapa a).

Esta metodologia de validação de protocolo é mostrada na Fig. 4.7. Nesta figura é mostrada também que o usuário pode obter o protocolo especificado em RP a partir de uma outra linguagem de projeto (como diagrama de estados, etc.) e fazer a sua validação. Neste caso o usuário deverá desenvolver as regras de conversão desta outra linguagem para RP.

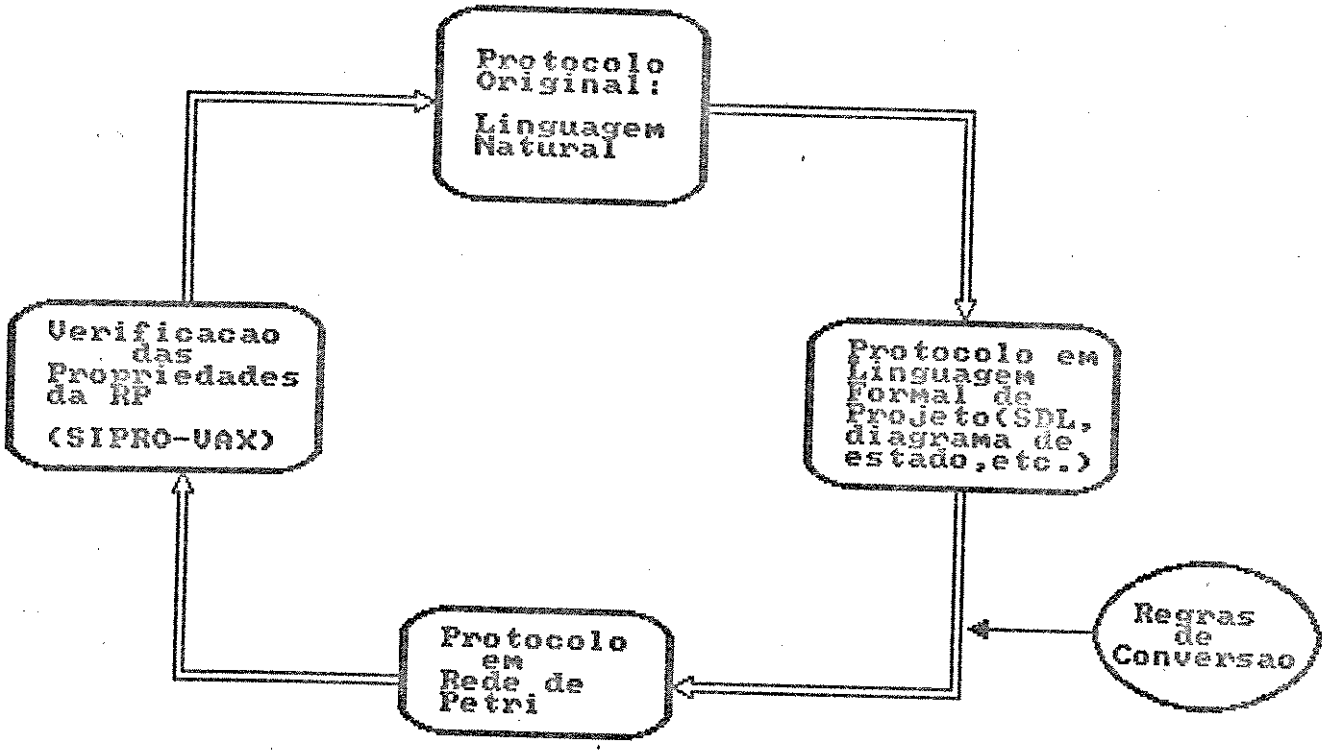


Fig. 4.7 - Metodologia de Validação com SIPRO-VAX.

Para a validação, do protocolo de sinalização entre os processadores da Trópico-R, foram consideradas as seguintes entidades envolvidas durante a transmissão de uma mensagem :

- Processador de origem (transmissor);
- IAP do plano A em modo de transmissão;
- IAP do plano B em modo de recepção;
- CVS do plano A;
- IAP do plano A em modo de recepção;
- Processador de destino (receptor).

OBS: - Nesta seção são adotados o termo sinal para indicar o fluxo de informações trocadas entre os processos, e o termo mensagem para indicar o pacote de dados trocados entre os processadores.

Os comportamentos dinâmicos destas entidades, durante a transmissão de mensagem, são representadas respectivamente por processo 1, 2, 3, 4, 5 e 6. Estes processos são separados em três blocos funcionais.

O bloco funcional 1 representa o módulo onde está localizado o processador (origem) que pode realizar a transmissão de uma mensagem no plano A, ou receber uma mensagem no plano B antes de enviar o segundo byte para o CVS do plano A.

O bloco funcional 2 representa o módulo de comutação do plano A, onde está localizado o CVS que está efetuando a varredura na IAP do módulo que está localizado o processador de origem.

O bloco funcional 3 representa o módulo onde está localizado o processador (destino) a qual o processador origem deseja transmitir uma mensagem. Este processador pode estar livre para a recepção ou ocupado, isto é, o processador de destino está transmitindo ou recebendo mensagem em outro plano.

Na Fig. 4.8 são mostradas os blocos funcionais com respectivos processos e o fluxo de sinais trocadas entre os processos.

A seguir, são apresentados os parâmetros dos sinis enviados de um processo para outro.

O processo 1 envia os seguintes sinais:

a) para o processo 2
   . Solicitação de Via-A
   . Não Solicita

b) para o processo 3
   . Solicitação de Via-B

c) para o processo 4
   . 1_Byte

. 2_Byte

d) para o processo 6
. Recepção OK
. Mensagem OK

O processo 1 não envia nenhum sinal para o processo 5, e recebe os seguintes sinais dos processos não considerados:

. Transmitir mensagem
. Mensagem OK

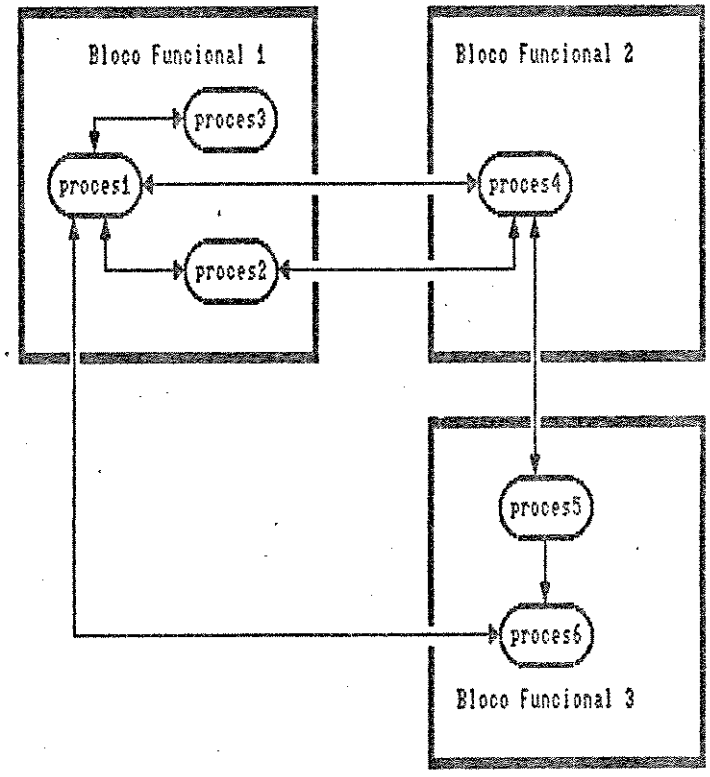


Fig. 4.8 - Blocos funcionais com respectivos processos.

O processo 2 envia os seguintes sinais:

a) para o processo 1
. Habilitação de transmissão

b) Para o processo 4
. Solicitação de vias

. Não solicita

O processo 3 envia o seguinte sinal:

a) para processo 1
   . Habilitação de recepção

b) este processo recebe os seguintes sinais de processo não especificado:
   . Habilitação de recepção B
   . Liberação de recepção

O processo 4 envia os seguintes sinais:

a) para o processo 1
   . 1_Byte OK
   . 1_Byte NOK

b) para processo 2
   . Habilitação de varredura
   . Habilitação de transmissão
   . Liberação de transmissão

c) para processo 5
   . Habilitação de recepção
   . Liberação de recepção

O processo 5 envia o seguinte sinal:

a) para processo 6
   . Habilitação de recepção

Finalmente, os sinais que o processo 6 envia são:

a) para o processo 1
   . Destino OK
   . Destino NOK
   . 2_Byte OK
   . 2_Byte NOK
   . Mensagem OK

As especificações dos processos 1, 2, 3, 4, 5 e 6 em diagrama SDL são mostradas respectivamente nas Fig. 4.9a, 4.9b, 4.9c, 4.10, 4.11a e 4.11b. Estas figuras apresentam também os códigos dos símbolos e de ramos utilizadas para a obtenção da

rede de Petri equivalente com o auxilio do programa SDLVAX.

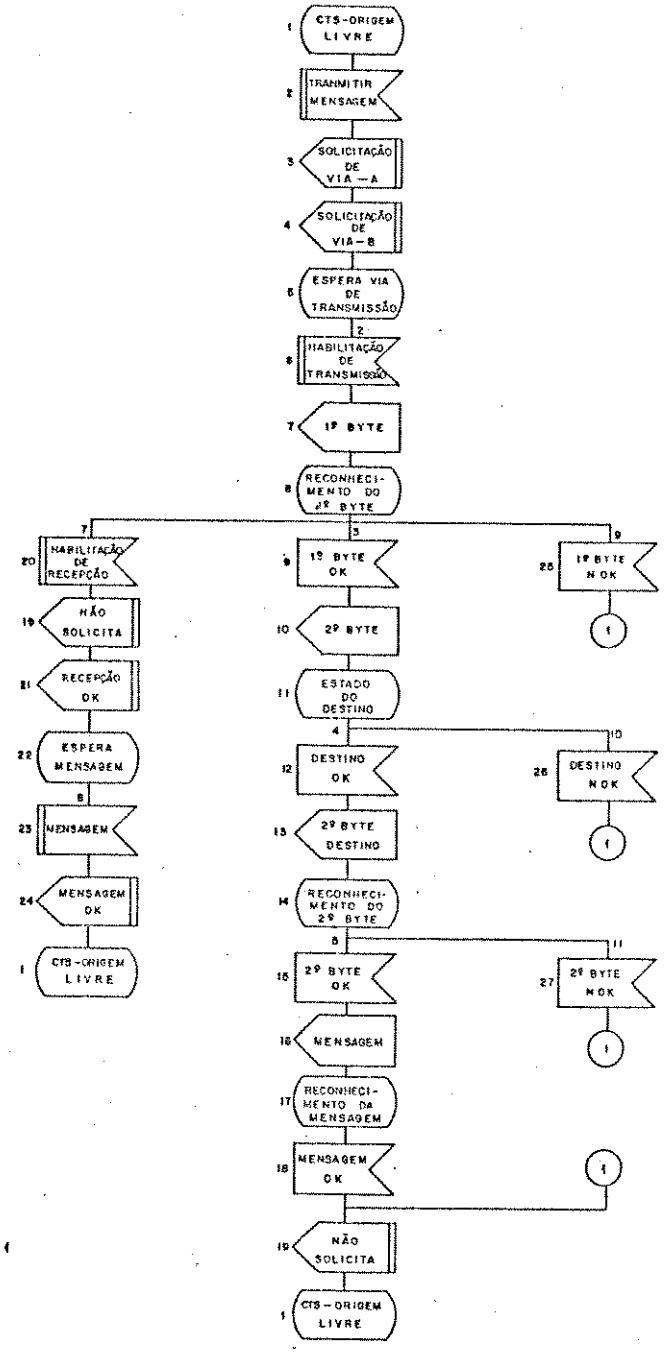
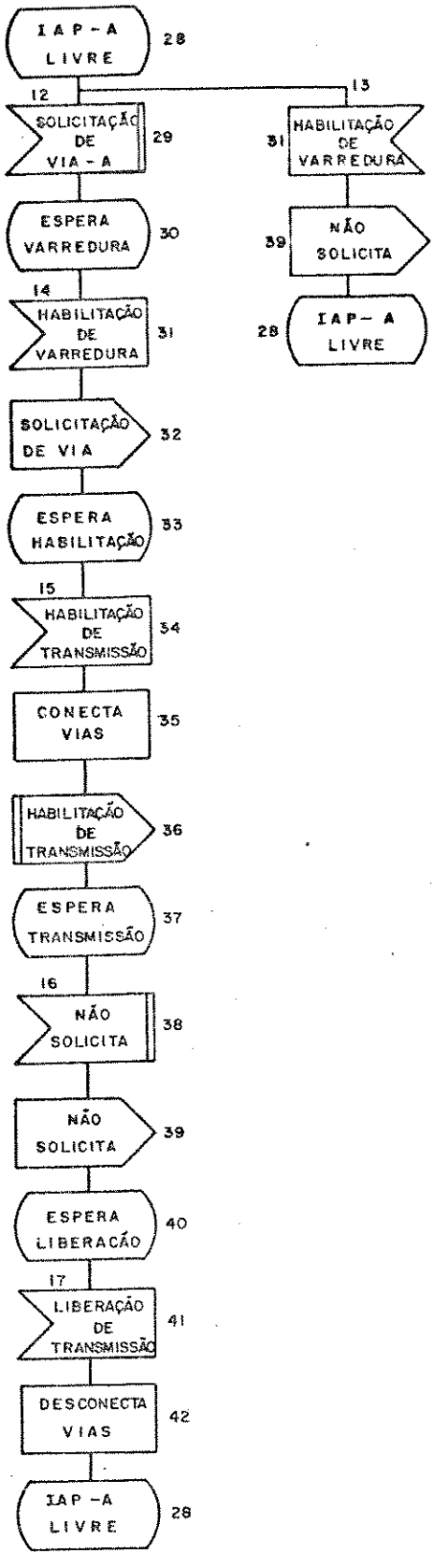
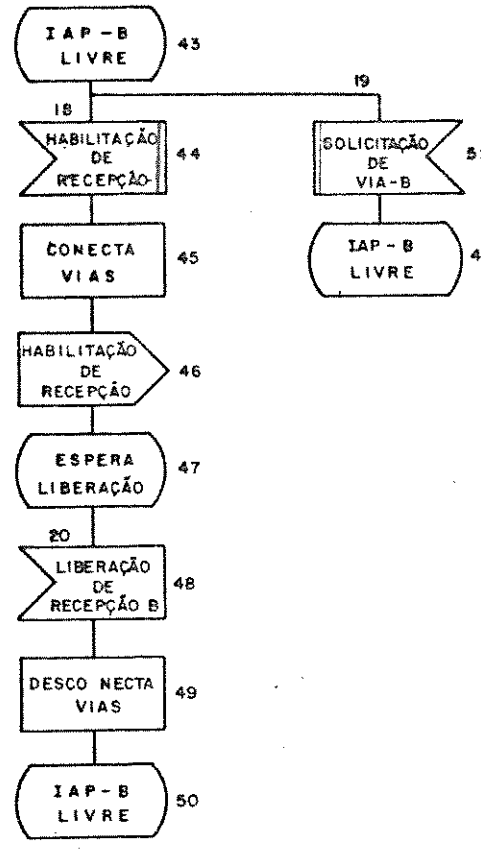


Fig. 4.9a – Processo do CTS de origem em SDL.



(b)



(c)

Fig 4.9b - Processo da IAP A do Bloc. Func. 1.
4.9c - Processo da IAP B do Bloc. Func. 1.

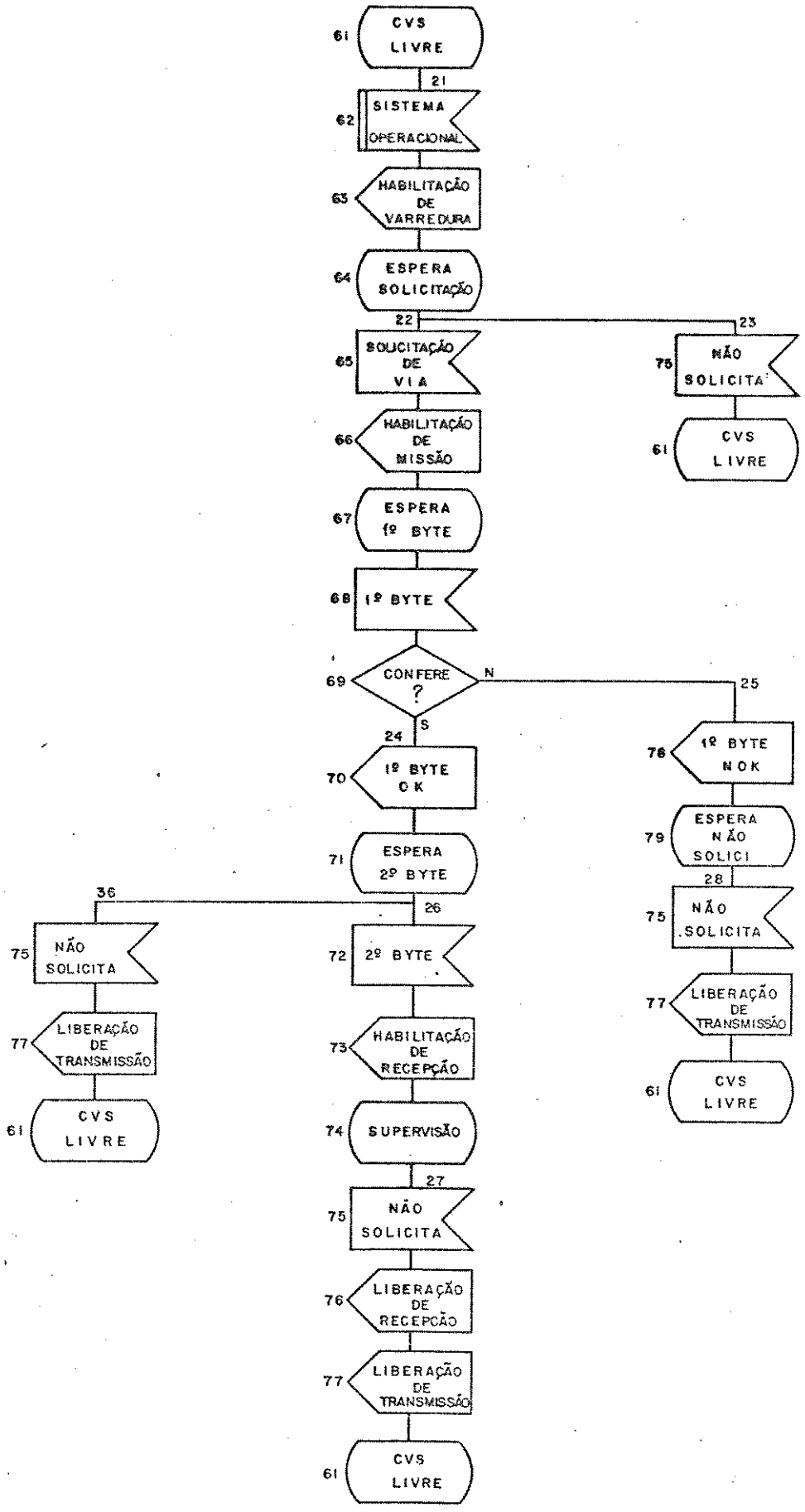


Fig 4.10 - Processo do CVS A em SDL.

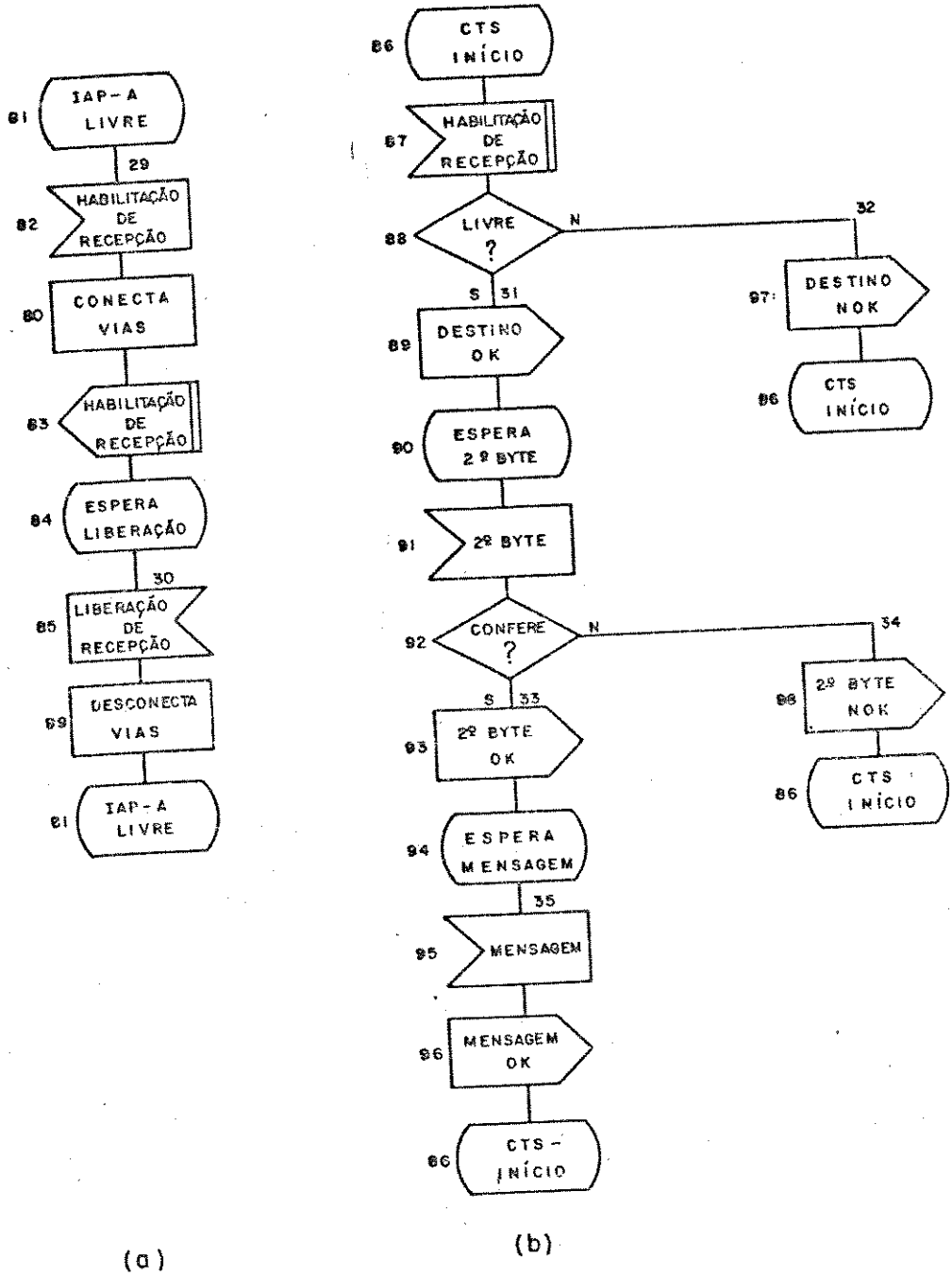


Fig. 4.11a - Processo da IAP A do CTS de Destino.
4.11b - Processo do CTS de Destino em SDL.

Após obter a rede de Petri do protocolo, foi feita a analise da verificação das propriedades da RP com o auxilio de ANAVAX. No resultado obtido, verificou-se que a propriedade de limitabilidade não foi satisfeita. A não verificação desta propriedade são devido a dois casos discutidos abaixo.

O primeiro caso de não limitabilidade ocorreu na parte da rede de Petri que representa o processo 3. Esta parte são formadas pelas barras 18, 19 e 20. A barra 18 apresenta apenas um lugar de entrada, pois o símbolo 44 não apresenta a representação na RP, mas envia as senhas nos lugares correspondentes dos símbolos 46 e 47.

Tendo uma senha no lugar correspondente do símbolo 47, a barra 20 fica habilitado a disparar, pois o símbolo 48 não apresenta um lugar correspondente na RP. E após o disparo da barra 20, o processo retorna ao estado inicial.

Consequentemente, as barras 18 e 20 podem disparar infinitas vezes, acumulando as senhas no lugar correspondente do símbolo 46.

Para resolver este caso, o processo 3 foi retirado da análise, pois da outra forma seria necessária a especificação do CVS do plano B para estes funcionarem em sincronismo, mas este tipo de interação é analisado entre os processos 4 e 5.

O segundo caso ocorreu devido a característica da concepção da chegada de sinal para um processo em SDL e na RP serem diferentes. Este caso foi verificado no processo 1 da Fig. 4.9a, quando o processo estando no estado "Reconhecimento do 1_Byte" (símbolo 8). Neste instante, ao receber o sinal "Habilitação de

Recepção" (símbolo 20) é realizada a transição 7 e fica no estado "Espera Mensagem" (símbolo 22). O processo neste estado, apenas o sinal "Mensagem" será reconhecido e realiza a transição 8. Se chegarem um sinal diferente deste, o sinal será consumido sem executar nenhuma transição. Isto é, estando no estado "Espera Mensagem", se chegar o sinal "1_Byte OK" ou "1_Byte NOK" proveniente do processo 4, estes sinais são consumidos automaticamente.

No caso da RP, o processo estando no lugar correspondente do símbolo 22 e se chegar uma senha no lugar correspondente do símbolo 9 ou 25 , as senhas destes lugares não serão consumidas, tornando a rede não limitada.

Para resolver este caso foram inseridos dois ramos após o ramo 7. A inclusão deste novos ramos são necessários simplesmente para satisfazer a propriedade de limitabilidade da rede de Petri, forçando a consumir as senhas do lugar correspondente do símbolo 9 (1_Byte OK) ou do símbolo 25 ("1_Byte NOK").

A especificação em diagrama SDL do novo processo 1 é apresentado na Fig. 4.12.

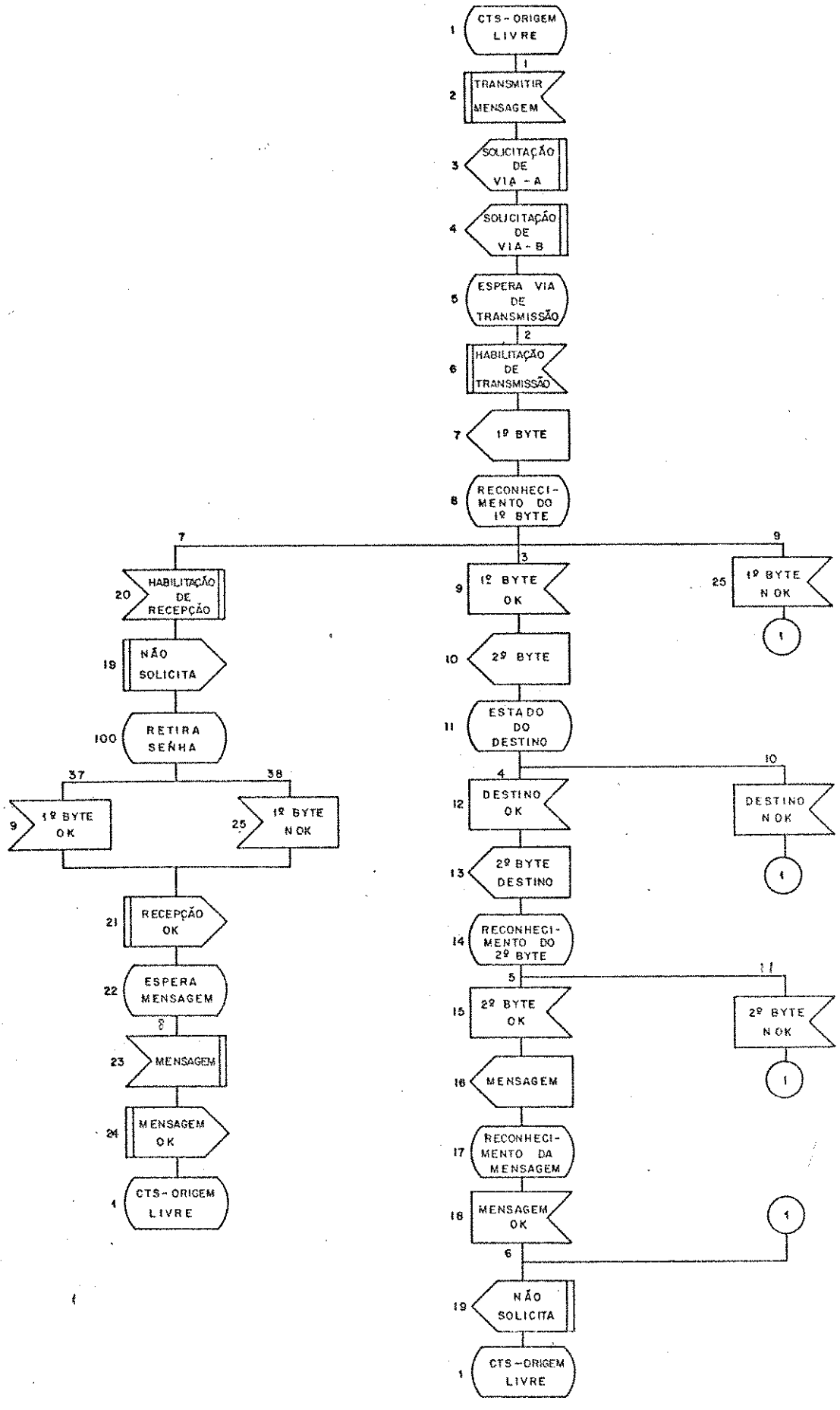


Fig. 4.12 - Processo do CTS de origem com modificações.

Para a segunda análise do protocolo, o bloco funcional 1 contém o processo 1 da Fig. 4.12 e processo 2 da Fig. 4.9b. Os blocos funcionais 2 e 3 serão os mesmos da análise anterior.

Os dados estruturais de rede de Petri obtida após a conversão estão apresentadas no Apendice 1.

Com auxilio de SIPRO-VAX, verificou-se que esta nova rede de Petri satisfez as propriedades de limitabilidade, vivacidade e reiniciabilidade. Isto é, nesta nova análise mostrou que o protocolo de comunicação entre processadores apresenta um funcionamento perfeito.

O resultado desta análise está, também apresentado no Apendice 1.

4.6 - Conclusão

Neste capítulo apresentou uma metodologia de validação de protocolos de comunicação. Baseada nesta metodologia, foi feito a validação do protocolo de sinalização entre os processadores da Trópico-R com o auxílio de SIPRO-VAX.

CAPÍTULO 5

CONCLUSÕES

Neste trabalho, foi apresentado um método formal de validação de protocolos de comunicação especificado em linguagem SDL. A metodologia consiste em transformar o protocolo descrito em SDL para a rede de Petri equivalente e verificar as suas propriedades.

Após estudos detalhados da linguagem SDL e da RP foi obtido um conjunto de regras simplificadas para a conversão da linguagem SDL para RP. AS regras obtidas (aplicáveis sòmente na validação de protocolos) foram bastante simples que permitiram a sua implementação automática em um computador.

Foi desenvolvido um pacote de "software" para auxiliar na validação do protocolo utilizando a metodologia apresentada. O pacote permite a um projetista entrar com um protocolo em SDL, realizar a sua conversão automática e analisar as propriedades da RP, também automaticamente.

Para ser possível o tratamento de protocolos de razoável complexidade, foi implementada também, a técnica de redução de RP apresentada em [3,5,13]. Esta técnica permite a verificação das propriedades da RP com um número reduzido de marcações.

Para verificar a funcionalidade da metodologia de validação apresentada, foi utilizado um exemplo de protocolo real. O exemplo utilizado foi o protocolo de sinalização entre processadores da central telefônica Trópico-R, desenvolvida pelo CPqD da Telebrás.

Verificou-se que embora a metodologia seja relativamente trabalhosa (metodologia iterativa) é possível validar protocolos de razoável complexidade.

Uma das limitacões do método de validação de protocolos discutido nesse trabalho, é o não tratamento de protocolos com temporizações. Isto porque a análise da rede de Petri é baseada em RP clássica. Assim, se forem desenvolvidos pacotes de "software" que permitam análise de RP temporizadas ou outro tipo mais completo de RP [14], o método de validação discutido neste trabalho, terá uma potencialidade bastante aumentada.

BIBLIOGRAFIAS

[1] CCITT - "Functional Specification and Description Language (SDL)" VOLUME VI - Fascicle VI.7 - Recomendações Z.101-104, 1980

[2] J.L.PETERSON - "Petri Net Theory and Modelling of Systens". Pretince - Hall, Inc., 1981

[3] G.BERTHELOT - "Verification des réseaux de Petri". These de 3éme Cycle, Univesité de Paris VI, Paris, 1978

[4] B.C.PRADIN - "Un Outil Graphique Interactif pour la Validation des Systens à Evolution Parallele Descrits par Reseaux de Petri". These de Docteur - Ingenieur, Université Paul Sabatier, Toulouse, 1979

[5] G.BERTHELOT, G.ROUCAIROL - "Reductin of Nets and Parallel Prograns". inc. proc. of Advanced Course on General Net Theory of Process and Systens. - Hamburg, 1979

[6] S.MOTOYAMA, W.C.BORELLI, R.T.TAMURA - "Validation of SDL - Comunication protocols by Means of Petri Nets" - TELECON - 1985.

[7] R.T.TAMURA, S.MOTOYAMA, W.C.BORELLI - "Conversor Automático de Protocolo de Comunicação em SDL para Rede de Petri" - quinto SBT - Setembro/1987

[8] S.MOTOYAMA,W.C.BORELLI,M.P.C.ARANTES,R.T.TAMURA - "Validação de Protocolos de Comunicação através de Rede de Petri : Aplicação em Sinalização telefônica" - quarto SBRC - março/1986

[9] S.MOTOYAMA,W.C.BORELLI,R.T.TAMURA - "Rede de Petri: Análise e Simulações" - RT-16, Publicação FEC 029/85

[10] W.J.COLLINS - "Programação Estruturada com Estudos de Casos em Pascal" - McGraw-Hill - 1986

[11] N.H.TANINAGA,R.T.TAMURA,S.BONATTI,S.MOTOYAMA - "Trópico-R: Modelagem de Filas e de Protocolos de Sinalização entre Processadores" - RT-24, Publicação FEC 029/86

[12] TELEBRÁS - "Trópico-R, Controle Local/Tanden Digital do Sistema Tropico - CPqD - Telebrás

[13] M.P.C. ARANTES - "Analisador Altomático de Rede de Petri para Validação de Protocolos de Comunicação " - Tese de Mestrado - FEE, UNICAMP, Fevereiro de 1988.

[14] F.J.W. SYMONS - "Introduction to Numerical Petri Nets, a General Graphical Model Concurrent Processing Systens" - Australian - Telecommunications Reserch, vol.14, No 1, 1980.

APENDICE 1

Dados Estruturais da Rede de Petri Correspondente do Protocolo de Sinalização entre Processadores da Trópico-R e Resultados da Análise Clássica.

DADOS ESTRUTURAIS DA REDE DE PETRI

MAIOR CODIGO DAS BARRAS = 38 QUANTIDADE DE BARRAS = 35
MAIOR CODIGO DOS LUGARES = 46 QUANTIDADE DE LUGARES = 46

BARRA - B1
ENTRADA - L 1 : 1
SAIDA - L 2 : 1 - L 3 : 1

BARRA - B2
ENTRADA - L 3 : 1 - L 4 : 1
SAIDA - L 5 : 1 - L 6 : 1

BARRA - B3
ENTRADA - L 6 : 1 - L 7 : 1
SAIDA - L 8 : 1 - L 9 : 1

BARRA - B4
ENTRADA - L 9 : 1 - L 10 : 1
SAIDA - L 11 : 1 - L 12 : 1

BARRA - B5
ENTRADA - L 12 : 1 - L 13 : 1
SAIDA - L 14 : 1 - L 15 : 1

BARRA - B6
ENTRADA - L 15 : 1 - L 16 : 1
SAIDA - L 1 : 1 - L 17 : 1

BARRA - B7
ENTRADA - L 6 : 1
SAIDA - L 17 : 1 - L 18 : 1

BARRA - B8
ENTRADA - L 19 : 1
SAIDA - L 1 : 1

BARRA - B9
ENTRADA - L 6 : 1 - L 20 : 1
SAIDA - L 1 : 1 - L 17 : 1

BARRA - B10
ENTRADA - L 9 : 1 - L 21 : 1
SAIDA - L 1 : 1 - L 17 : 1

BARRA - B11
ENTRADA - L 12 : 1 - L 22 : 1
SAIDA - L 1 : 1 - L 17 : 1

BARRA - B12
ENTRADA - L 2 : 1 - L 23 : 1
SAIDA - L 24 : 1

```
BARRA - B13
ENTRADA - L 23 :  1 - L 25 :  1
SAIDA   - L 23 :  1 - L 26 :  1

BARRA - B14
ENTRADA - L 24 :  1 - L 25 :  1
SAIDA   - L 27 :  1 - L 28 :  1

BARRA - B15
ENTRADA - L 28 :  1 - L 29 :  1
SAIDA   - L  4 :  1 - L 30 :  1

BARRA - B16
ENTRADA - L 17 :  1 - L 30 :  1
SAIDA   - L 26 :  1 - L 31 :  1

BARRA - B17
ENTRADA - L 31 :  1 - L 32 :  1
SAIDA   - L 23 :  1

BARRA - B21
ENTRADA - L 33 :  1
SAIDA   - L 25 :  1 - L 34 :  1

BARRA - B22
ENTRADA - L 27 :  1 - L 34 :  1
SAIDA   - L 29 :  1 - L 35 :  1

BARRA - B23
ENTRADA - L 26 :  1 - L 34 :  1
SAIDA   - L 33 :  1

BARRA - B24
ENTRADA - L  5 :  1 - L 35 :  1
SAIDA   - L  7 :  1 - L 36 :  1

BARRA - B25
ENTRADA - L  5 :  1 - L 35 :  1
SAIDA   - L 20 :  1 - L 37 :  1

BARRA - B26
ENTRADA - L  8 :  1 - L 36 :  1
SAIDA   - L 38 :  1 - L 39 :  1

BARRA - B27
ENTRADA - L 26 :  1 - L 39 :  1
SAIDA   - L 32 :  1 - L 33 :  1 - L 40 :  1

BARRA - B28
ENTRADA - L 26 :  1 - L 37 :  1
SAIDA   - L 32 :  1 - L 33 :  1

BARRA - B29
ENTRADA - L 38 :  1 - L 41 :  1
SAIDA   - L 42 :  1 - L 43 :  1
```

```
BARRA - B30
ENTRADA - L 40 :  1 - L 43 :  1
SAIDA   - L 41 :  1

BARRA - B31
ENTRADA - L 42 :  1 - L 44 :  1
SAIDA   - L 10 :  1 - L 45 :  1

BARRA - B32
ENTRADA - L 42 :  1 - L 44 :  1
SAIDA   - L 21 :  1 - L 44 :  1

BARRA - B33
ENTRADA - L 11 :  1 - L 45 :  1
SAIDA   - L 13 :  1 - L 46 :  1

BARRA - B34
ENTRADA - L 11 :  1 - L 45 :  1
SAIDA   - L 22 :  1 - L 44 :  1

BARRA - B35
ENTRADA - L 14 :  1 - L 46 :  1
SAIDA   - L 16 :  1 - L 44 :  1

BARRA - B36
ENTRADA - L 26 :  1 - L 36 :  1
SAIDA   - L 32 :  1 - L 33 :  1

BARRA - B37
ENTRADA - L  7 :  1 - L 18 :  1
SAIDA   - L 19 :  1

BARRA - B38
ENTRADA - L 18 :  1 - L 20 :  1
SAIDA   - L 19 :  1


MARCACAO INICIAL

LUGARES - L  1 :  1 - L 23 :  1 - L 33 :  1
        - L 41 :  1 - L 44 :  1
```

DOCUMENTACAO DE NOMES DOS LUGARES

```
LUGAR L  1  NOME =  CTS - Origem Livre
LUGAR L  2  NOME =  Solicitacao de Via - A
LUGAR L  3  NOME =  Espera Via de Transmissao
LUGAR L  4  NOME =  Habilitacao de Transmissao
LUGAR L  5  NOME =  1_Byte
LUGAR L  6  NOME =  Reconhecimento do 1_Byte
LUGAR L  7  NOME =  1_Byte OK
LUGAR L  8  NOME =  2_Byte
LUGAR L  9  NOME =  Estado do Destino
LUGAR L 10  NOME =  Destino OK
LUGAR L 11  NOME =  2_Byte Destino
LUGAR L 12  NOME =  Reconhecimento do 2_Byte
LUGAR L 13  NOME =  2_Byte OK
LUGAR L 14  NOME =  Mensagem
LUGAR L 15  NOME =  Reconhecimento de Mensagem
LUGAR L 16  NOME =  Mensagem OK
LUGAR L 17  NOME =  Nao Solicita
LUGAR L 18  NOME =  Retira Senha
LUGAR L 19  NOME =  Espera Mensagem
LUGAR L 20  NOME =  1_Byte NOK
LUGAR L 21  NOME =  Destino NOK
LUGAR L 22  NOME =  2_Byte NOK
LUGAR L 23  NOME =  IAP_A Livre
LUGAR L 24  NOME =  Espera Varredura
LUGAR L 25  NOME =  Habilitacao de Varredura
LUGAR L 26  NOME =  Nao Solicita
LUGAR L 27  NOME =  Solicitacao de Via
LUGAR L 28  NOME =  Espera Habilitacao
LUGAR L 29  NOME =  Habilitacao de Transmissao
LUGAR L 30  NOME =  Espera Transmissao
LUGAR L 31  NOME =  Espera Liberacao
LUGAR L 32  NOME =  Liberacao de Transmissao
LUGAR L 33  NOME =  CVS Livre
LUGAR L 34  NOME =  Espera Solicitacao
LUGAR L 35  NOME =  Espera 1_Byte
LUGAR L 36  NOME =  Espera 2_Byte
LUGAR L 37  NOME =  Espera Nao Solicita
LUGAR L 38  NOME =  Habilitacao de Recepcao
LUGAR L 39  NOME =  Supervisao
LUGAR L 40  NOME =  Liberacao de Recepcao
LUGAR L 41  NOME =  IAP_A_Livre
LUGAR L 42  NOME =  Habilitacao de Recepcao
LUGAR L 43  NOME =  Espera Liberacao
LUGAR L 44  NOME =  CTS Destino Inicio
LUGAR L 45  NOME =  Espera 2_Byte
LUGAR L 46  NOME =  Espera Mensagem
```

ANÁLISE DA REDE DE PETRI - DOCUMENTAÇÃO

Número Total de Marcações : 120

REDE Limitada, Reiniciável e Viva.

MAQUINA DE SENHA

(Marcação :: Barra que Dispara -> Próxima Marcação)

M	0	::	B	21	-> M	2	B	1	-> M	1	
M	1	::	B	12	-> M	3	B	21	-> M	4	
M	2	::	B	1	-> M	4	B	13	-> M	5	
M	3	::	B	21	-> M	6					
M	4	::	B	12	-> M	6	B	13	-> M	7	
M	5	::	B	23	-> M	0	B	1	-> M	7	
M	6	::	B	14	-> M	8					
M	7	::	B	23	-> M	1	B	12	-> M	9	
M	8	::	B	22	-> M	10					
M	9	::	B	23	-> M	3					
M	10	::	B	15	-> M	11					
M	11	::	B	2	-> M	12					
M	12	::	B	7	-> M	13	B	24	-> M	14	B 25 -> M 15
M	13	::	B	16	-> M	16	B	24	-> M	17	B 25 -> M 18
M	14	::	B	7	-> M	17	B	3	-> M	19	
M	15	::	B	7	-> M	18	B	9	-> M	20	
M	16	::	B	24	-> M	21	B	25	-> M	22	
M	17	::	B	16	-> M	21	B	37	-> M	23	
M	18	::	B	16	-> M	22	B	38	-> M	24	
M	19	::	B	26	-> M	25					
M	20	::	B	1	-> M	26	B	16	-> M	27	
M	21	::	B	36	-> M	28	B	37	-> M	29	
M	22	::	B	28	-> M	30	B	38	-> M	31	
M	23	::	B	16	-> M	29	B	8	-> M	32	
M	24	::	B	8	-> M	20	B	16	-> M	31	
M	25	::	B	29	-> M	33					
M	26	::	B	16	-> M	34					
M	27	::	B	1	-> M	34	B	28	-> M	35	
M	28	::	B	17	-> M	36	B	21	-> M	37	B 37 -> M 38
M	29	::	B	36	-> M	38	B	8	-> M	39	
M	30	::	B	38	-> M	38	B	17	-> M	40	B 21 -> M 41
M	31	::	B	8	-> M	27	B	28	-> M	38	
M	32	::	B	16	-> M	39	B	1	-> M	42	
M	33	::	B	31	-> M	43	B	32	-> M	44	
M	34	::	B	28	-> M	45					
M	35	::	B	17	-> M	0	B	1	-> M	45	B 21 -> M 46
M	36	::	B	21	-> M	47	B	37	-> M	48	
M	37	::	B	17	-> M	47	B	37	-> M	49	
M	38	::	B	8	-> M	35	B	17	-> M	48	B 21 -> M 49
M	39	::	B	36	-> M	35	B	1	-> M	50	
M	40	::	B	38	-> M	48	B	21	-> M	51	
M	41	::	B	38	-> M	49	B	17	-> M	51	

```
M  42 ::      B 16 -> M  50
M  43 ::      B  4 -> M  52
M  44 ::      B 10 -> M  53
M  45 ::      B 17 -> M   1      B 21 -> M  54
M  46 ::      B 17 -> M   2      B  1 -> M  54
M  47 ::      B 13 -> M  55      B 37 -> M  56
M  48 ::      B  8 -> M   0      B 21 -> M  56
M  49 ::      B  8 -> M  46      B 17 -> M  56
M  50 ::      B 36 -> M  45
M  51 ::      B 38 -> M  56      B 13 -> M  57
M  52 ::      B 33 -> M  58      B 34 -> M  59
M  53 ::      B  1 -> M  60      B 16 -> M  61
M  54 ::      B 17 -> M   4
M  55 ::      B 23 -> M  36      B 37 -> M  62
M  56 ::      B  8 -> M   2      B 13 -> M  62
M  57 ::      B 23 -> M  40      B 38 -> M  62
M  58 ::      B  5 -> M  63
M  59 ::      B 11 -> M  53
M  60 ::      B 16 -> M  64
M  61 ::      B  1 -> M  64      B 27 -> M  65
M  62 ::      B  8 -> M   5      B 23 -> M  48
M  63 ::      B 35 -> M  66
M  64 ::      B 27 -> M  67
M  65 ::      B 30 -> M  35      B  1 -> M  67      B 17 -> M  68
              B 21 -> M  69
M  66 ::      B  6 -> M  53
M  67 ::      B 30 -> M  45      B 17 -> M  70      B 21 -> M  71
M  68 ::      B 30 -> M   0      B  1 -> M  70      B 21 -> M  72
M  69 ::      B 30 -> M  46      B  1 -> M  71      B 17 -> M  72
M  70 ::      B 30 -> M   1      B 12 -> M  73      B 21 -> M  74
M  71 ::      B 30 -> M  54      B 17 -> M  74
M  72 ::      B 30 -> M   2      B  1 -> M  74      B 13 -> M  75
M  73 ::      B 30 -> M   3      B 21 -> M  76
M  74 ::      B 30 -> M   4      B 12 -> M  76      B 13 -> M  77
M  75 ::      B 30 -> M   5      B 23 -> M  68      B  1 -> M  77
M  76 ::      B 30 -> M   6      B 14 -> M  78
M  77 ::      B 30 -> M   7      B 23 -> M  70      B 12 -> M  79
M  78 ::      B 30 -> M   8      B 22 -> M  80
M  79 ::      B 30 -> M   9      B 23 -> M  73
M  80 ::      B 30 -> M  10      B 15 -> M  81
M  81 ::      B 30 -> M  11      B  2 -> M  82
M  82 ::      B 30 -> M  12      B  7 -> M  83      B 24 -> M  84
              B 25 -> M  85
M  83 ::      B 30 -> M  13      B 16 -> M  86      B 24 -> M  87
              B 25 -> M  88
M  84 ::      B 30 -> M  14      B  7 -> M  87      B  3 -> M  89
M  85 ::      B 30 -> M  15      B  7 -> M  88      B  9 -> M  90
M  86 ::      B 30 -> M  16      B 24 -> M  91      B 25 -> M  92
M  87 ::      B 30 -> M  17      B 16 -> M  91      B 37 -> M  93
M  88 ::      B 30 -> M  18      B 16 -> M  92      B 38 -> M  94
M  89 ::      B 30 -> M  19      B 26 -> M  95
M  90 ::      B 30 -> M  20      B  1 -> M  96      B 16 -> M  97
M  91 ::      B 30 -> M  21      B 36 -> M  98      B 37 -> M  99
M  92 ::      B 30 -> M  22      B 28 -> M 100      B 38 -> M 101
M  93 ::      B 30 -> M  23      B 16 -> M  99      B  8 -> M 102
```

```
M  94 ::      B 30 -> M  24      B  8 -> M  90      B 16 -> M 101
M  95 ::      B 30 -> M  25
M  96 ::      B 30 -> M  26      B 16 -> M 103
M  97 ::      B 30 -> M  27      B 28 -> M  65      B  1 -> M 103
M  98 ::      B 30 -> M  28      B 17 -> M 104      B 21 -> M 105
              B 37 -> M 106
M  99 ::      B 30 -> M  29      B 36 -> M 106      B  8 -> M 107
M 100 ::      B 30 -> M  30      B 38 -> M 106      B 17 -> M 108
              B 21 -> M 109
M 101 ::      B 30 -> M  31      B  8 -> M  97      B 28 -> M 106
M 102 ::      B 30 -> M  32      B 16 -> M 107      B  1 -> M 110
M 103 ::      B 30 -> M  34      B 28 -> M  67
M 104 ::      B 30 -> M  36      B 21 -> M 111      B 37 -> M 112
M 105 ::      B 30 -> M  37      B 17 -> M 111      B 37 -> M 113
M 106 ::      B 30 -> M  38      B  8 -> M  65      B 17 -> M 112
              B 21 -> M 113
M 107 ::      B 30 -> M  39      B 36 -> M  65      B  1 -> M 114
M 108 ::      B 30 -> M  40      B 38 -> M 112      B 21 -> M 115
M 109 ::      B 30 -> M  41      B 38 -> M 113      B 17 -> M 115
M 110 ::      B 30 -> M  42      B 16 -> M 114
M 111 ::      B 30 -> M  47      B 13 -> M 116      B 37 -> M 117
M 112 ::      B 30 -> M  48      B  8 -> M  68      B 21 -> M 117
M 113 ::      B 30 -> M  49      B  8 -> M  69      B 17 -> M 117
M 114 ::      B 30 -> M  50      B 36 -> M  67
M 115 ::      B 30 -> M  51      B 38 -> M 117      B 13 -> M 118
M 116 ::      B 30 -> M  55      B 23 -> M 104      B 37 -> M 119
M 117 ::      B 30 -> M  56      B  8 -> M  72      B 13 -> M 119
M 118 ::      B 30 -> M  57      B 23 -> M 108      B 38 -> M 119
M 119 ::      B 30 -> M  62      B  8 -> M  75      B 23 -> M 112
```

TABELA DE MARCAÇÕES

(Marcação -> Lugar : Número de Senhas)

M 0 ->	L 1 : 1	L 23 : 1	L 33 : 1	L 41 : 1	
	L 44 : 1				
M 1 ->	L 2 : 1	L 3 : 1	L 23 : 1	L 33 : 1	
	L 41 : 1	L 44 : 1			
M 2 ->	L 1 : 1	L 23 : 1	L 25 : 1	L 34 : 1	
	L 41 : 1	L 44 : 1			
M 3 ->	L 3 : 1	L 24 : 1	L 33 : 1	L 41 : 1	
	L 44 : 1				
M 4 ->	L 2 : 1	L 3 : 1	L 23 : 1	L 25 : 1	
	L 34 : 1	L 41 : 1	L 44 : 1		
M 5 ->	L 1 : 1	L 23 : 1	L 26 : 1	L 34 : 1	
	L 41 : 1	L 44 : 1			
M 6 ->	L 3 : 1	L 24 : 1	L 25 : 1	L 34 : 1	
	L 41 : 1	L 44 : 1			
M 7 ->	L 2 : 1	L 3 : 1	L 23 : 1	L 26 : 1	
	L 34 : 1	L 41 : 1	L 44 : 1		
M 8 ->	L 3 : 1	L 27 : 1	L 28 : 1	L 34 : 1	
	L 41 : 1	L 44 : 1			
M 9 ->	L 3 : 1	L 24 : 1	L 26 : 1	L 34 : 1	
	L 41 : 1	L 44 : 1			
M 10 ->	L 3 : 1	L 28 : 1	L 29 : 1	L 35 : 1	
	L 41 : 1	L 44 : 1			
M 11 ->	L 3 : 1	L 4 : 1	L 30 : 1	L 35 : 1	
	L 41 : 1	L 44 : 1			
M 12 ->	L 5 : 1	L 6 : 1	L 30 : 1	L 35 : 1	
	L 41 : 1	L 44 : 1			
M 13 ->	L 5 : 1	L 17 : 1	L 18 : 1	L 30 : 1	
	L 35 : 1	L 41 : 1	L 44 : 1		
M 14 ->	L 6 : 1	L 7 : 1	L 30 : 1	L 36 : 1	
	L 41 : 1	L 44 : 1			
M 15 ->	L 6 : 1	L 20 : 1	L 30 : 1	L 37 : 1	
	L 41 : 1	L 44 : 1			
M 16 ->	L 5 : 1	L 18 : 1	L 26 : 1	L 31 : 1	
	L 35 : 1	L 41 : 1	L 44 : 1		
M 17 ->	L 7 : 1	L 17 : 1	L 18 : 1	L 30 : 1	
	L 36 : 1	L 41 : 1	L 44 : 1		
M 18 ->	L 17 : 1	L 18 : 1	L 20 : 1	L 30 : 1	
	L 37 : 1	L 41 : 1	L 44 : 1		
M 19 ->	L 8 : 1	L 9 : 1	L 30 : 1	L 36 : 1	
	L 41 : 1	L 44 : 1			
M 20 ->	L 1 : 1	L 17 : 1	L 30 : 1	L 37 : 1	
	L 41 : 1	L 44 : 1			
M 21 ->	L 7 : 1	L 18 : 1	L 26 : 1	L 31 : 1	
	L 36 : 1	L 41 : 1	L 44 : 1		
M 22 ->	L 18 : 1	L 20 : 1	L 26 : 1	L 31 : 1	
	L 37 : 1	L 41 : 1	L 44 : 1		
M 23 ->	L 17 : 1	L 19 : 1	L 30 : 1	L 36 : 1	
	L 41 : 1	L 44 : 1			
M 24 ->	L 17 : 1	L 19 : 1	L 30 : 1	L 37 : 1	
	L 41 : 1	L 44 : 1			

M 25 ->
L 9 : 1
L 41 : 1
L 30 : 1
L 44 : 1
L 38 : 1
L 39 : 1

M 26 ->
L 2 : 1
L 37 : 1
L 3 : 1
L 41 : 1
L 17 : 1
L 44 : 1
L 30 : 1

M 27 ->
L 1 : 1
L 41 : 1
L 26 : 1
L 44 : 1
L 31 : 1
L 37 : 1

M 28 ->
L 7 : 1
L 33 : 1
L 18 : 1
L 41 : 1
L 31 : 1
L 44 : 1
L 32 : 1

M 29 ->
L 19 : 1
L 41 : 1
L 26 : 1
L 44 : 1
L 31 : 1
L 36 : 1

M 30 ->
L 18 : 1
L 33 : 1
L 20 : 1
L 41 : 1
L 31 : 1
L 44 : 1
L 32 : 1

M 31 ->
L 19 : 1
L 41 : 1
L 26 : 1
L 44 : 1
L 31 : 1
L 37 : 1

M 32 ->
L 1 : 1
L 41 : 1
L 17 : 1
L 44 : 1
L 30 : 1
L 36 : 1

M 33 ->
L 9 : 1
L 43 : 1
L 30 : 1
L 44 : 1
L 39 : 1
L 42 : 1

M 34 ->
L 2 : 1
L 37 : 1
L 3 : 1
L 41 : 1
L 26 : 1
L 44 : 1
L 31 : 1

M 35 ->
L 1 : 1
L 41 : 1
L 31 : 1
L 44 : 1
L 32 : 1
L 33 : 1

M 36 ->
L 7 : 1
L 41 : 1
L 18 : 1
L 44 : 1
L 23 : 1
L 33 : 1

M 37 ->
L 7 : 1
L 32 : 1
L 18 : 1
L 34 : 1
L 25 : 1
L 41 : 1
L 31 : 1
L 44 : 1

M 38 ->
L 19 : 1
L 41 : 1
L 31 : 1
L 44 : 1
L 32 : 1
L 33 : 1

M 39 ->
L 1 : 1
L 41 : 1
L 26 : 1
L 44 : 1
L 31 : 1
L 36 : 1

M 40 ->
L 18 : 1
L 41 : 1
L 20 : 1
L 44 : 1
L 23 : 1
L 33 : 1

M 41 ->
L 18 : 1
L 32 : 1
L 20 : 1
L 34 : 1
L 25 : 1
L 41 : 1
L 31 : 1
L 44 : 1

M 42 ->
L 2 : 1
L 36 : 1
L 3 : 1
L 41 : 1
L 17 : 1
L 44 : 1
L 30 : 1

M 43 ->
L 9 : 1
L 43 : 1
L 10 : 1
L 45 : 1
L 30 : 1
L 38 : 1

M 44 ->
L 9 : 1
L 43 : 1
L 21 : 1
L 44 : 1
L 30 : 1
L 39 : 1

M 45 ->
L 2 : 1
L 33 : 1
L 3 : 1
L 41 : 1
L 31 : 1
L 44 : 1
L 32 : 1

M 46 ->
L 1 : 1
L 34 : 1
L 25 : 1
L 41 : 1
L 31 : 1
L 44 : 1
L 32 : 1

M 47 ->
L 7 : 1
L 34 : 1
L 18 : 1
L 41 : 1
L 23 : 1
L 44 : 1
L 25 : 1

M 48 ->
L 19 : 1
L 44 : 1
L 23 : 1
L 41 : 1
L 33 : 1
L 41 : 1

M 49 ->
L 19 : 1
L 34 : 1
L 25 : 1
L 41 : 1
L 31 : 1
L 44 : 1
L 32 : 1

M 50 ->
L 2 : 1
L 36 : 1
L 3 : 1
L 41 : 1
L 26 : 1
L 44 : 1
L 31 : 1

M 51 ->
L 18 : 1
L 34 : 1
L 20 : 1
L 41 : 1
L 23 : 1
L 44 : 1
L 25 : 1

M 52 -> L 11 : 1 L 12 : 1 L 30 : 1 L 39 : 1
L 43 : 1 L 45 : 1
M 53 -> L 1 : 1 L 17 : 1 L 30 : 1 L 39 : 1
L 43 : 1 L 44 : 1
M 54 -> L 2 : 1 L 3 : 1 L 25 : 1 L 31 : 1
L 32 : 1 L 34 : 1 L 41 : 1 L 44 : 1
M 55 -> L 7 : 1 L 18 : 1 L 23 : 1 L 26 : 1
L 34 : 1 L 41 : 1 L 44 : 1
M 56 -> L 19 : 1 L 23 : 1 L 25 : 1 L 34 : 1
L 41 : 1 L 44 : 1
M 57 -> L 18 : 1 L 20 : 1 L 23 : 1 L 26 : 1
L 34 : 1 L 41 : 1 L 44 : 1
M 58 -> L 12 : 1 L 13 : 1 L 30 : 1 L 39 : 1
L 43 : 1 L 46 : 1
M 59 -> L 12 : 1 L 22 : 1 L 30 : 1 L 39 : 1
L 43 : 1 L 44 : 1
M 60 -> L 2 : 1 L 3 : 1 L 17 : 1 L 30 : 1
L 39 : 1 L 43 : 1 L 44 : 1
M 61 -> L 1 : 1 L 26 : 1 L 31 : 1 L 39 : 1
L 43 : 1 L 44 : 1
M 62 -> L 19 : 1 L 23 : 1 L 26 : 1 L 34 : 1
L 41 : 1 L 44 : 1
M 63 -> L 14 : 1 L 15 : 1 L 30 : 1 L 39 : 1
L 43 : 1 L 46 : 1
M 64 -> L 2 : 1 L 3 : 1 L 26 : 1 L 31 : 1
L 39 : 1 L 43 : 1 L 44 : 1
M 65 -> L 1 : 1 L 31 : 1 L 32 : 1 L 33 : 1
L 40 : 1 L 43 : 1 L 44 : 1
M 66 -> L 15 : 1 L 16 : 1 L 30 : 1 L 39 : 1
L 43 : 1 L 44 : 1
M 67 -> L 2 : 1 L 3 : 1 L 31 : 1 L 32 : 1
L 33 : 1 L 40 : 1 L 43 : 1 L 44 : 1
M 68 -> L 1 : 1 L 23 : 1 L 33 : 1 L 40 : 1
L 43 : 1 L 44 : 1
M 69 -> L 1 : 1 L 25 : 1 L 31 : 1 L 32 : 1
L 34 : 1 L 40 : 1 L 43 : 1 L 44 : 1
M 70 -> L 2 : 1 L 3 : 1 L 23 : 1 L 33 : 1
L 40 : 1 L 43 : 1 L 44 : 1
M 71 -> L 2 : 1 L 3 : 1 L 25 : 1 L 31 : 1
L 32 : 1 L 34 : 1 L 40 : 1 L 43 : 1
L 44 : 1
M 72 -> L 1 : 1 L 23 : 1 L 25 : 1 L 34 : 1
L 40 : 1 L 43 : 1 L 44 : 1
M 73 -> L 3 : 1 L 24 : 1 L 33 : 1 L 40 : 1
L 43 : 1 L 44 : 1
M 74 -> L 2 : 1 L 3 : 1 L 23 : 1 L 25 : 1
L 34 : 1 L 40 : 1 L 43 : 1 L 44 : 1
M 75 -> L 1 : 1 L 23 : 1 L 26 : 1 L 34 : 1
L 40 : 1 L 43 : 1 L 44 : 1
M 76 -> L 3 : 1 L 24 : 1 L 25 : 1 L 34 : 1
L 40 : 1 L 43 : 1 L 44 : 1
M 77 -> L 2 : 1 L 3 : 1 L 23 : 1 L 26 : 1
L 34 : 1 L 40 : 1 L 43 : 1 L 44 : 1
M 78 -> L 3 : 1 L 27 : 1 L 28 : 1 L 34 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 79 -> L 3 : 1 L 24 : 1 L 26 : 1 L 34 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 80 -> L 3 : 1 L 28 : 1 L 29 : 1 L 35 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 81 -> L 3 : 1 L 4 : 1 L 30 : 1 L 35 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 82 -> L 5 : 1 L 6 : 1 L 30 : 1 L 35 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 83 -> L 5 : 1 L 17 : 1 L 18 : 1 L 30 : 1
L 35 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 84 -> L 6 : 1 L 7 : 1 L 30 : 1 L 36 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 85 -> L 6 : 1 L 20 : 1 L 30 : 1 L 37 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 86 -> L 5 : 1 L 18 : 1 L 26 : 1 L 31 : 1
L 35 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 87 -> L 7 : 1 L 17 : 1 L 18 : 1 L 30 : 1
L 36 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 88 -> L 17 : 1 L 18 : 1 L 20 : 1 L 30 : 1
L 37 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 89 -> L 8 : 1 L 9 : 1 L 30 : 1 L 36 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 90 -> L 1 : 1 L 17 : 1 L 30 : 1 L 37 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 91 -> L 7 : 1 L 18 : 1 L 26 : 1 L 31 : 1
L 36 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 92 -> L 18 : 1 L 20 : 1 L 26 : 1 L 31 : 1
L 37 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 93 -> L 17 : 1 L 19 : 1 L 30 : 1 L 36 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 94 -> L 17 : 1 L 19 : 1 L 30 : 1 L 37 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 95 -> L 9 : 1 L 30 : 1 L 38 : 1 L 39 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 96 -> L 2 : 1 L 3 : 1 L 17 : 1 L 30 : 1
L 37 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 97 -> L 1 : 1 L 26 : 1 L 31 : 1 L 37 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 98 -> L 7 : 1 L 18 : 1 L 31 : 1 L 32 : 1
L 33 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 99 -> L 19 : 1 L 26 : 1 L 31 : 1 L 36 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 100 -> L 18 : 1 L 20 : 1 L 31 : 1 L 32 : 1
L 33 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 101 -> L 19 : 1 L 26 : 1 L 31 : 1 L 37 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 102 -> L 1 : 1 L 17 : 1 L 30 : 1 L 36 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 103 -> L 2 : 1 L 3 : 1 L 26 : 1 L 31 : 1
L 37 : 1 L 40 : 1 L 43 : 1 L 44 : 1

M 104 -> L 7 : 1 L 18 : 1 L 23 : 1 L 33 : 1
L 40 : 1 L 43 : 1 L 44 : 1

M 105 -> L 7 : 1 L 18 : 1 L 25 : 1 L 31 : 1
L 32 : 1 L 34 : 1 L 40 : 1 L 43 : 1
L 44 : 1

M 106 ->	L 19 : 1	L 31 : 1	L 32 : 1	L 33 : 1
	L 40 : 1	L 43 : 1	L 44 : 1	
M 107 ->	L 1 : 1	L 26 : 1	L 31 : 1	L 36 : 1
	L 40 : 1	L 43 : 1	L 44 : 1	
M 108 ->	L 18 : 1	L 20 : 1	L 23 : 1	L 33 : 1
	L 40 : 1	L 43 : 1	L 44 : 1	
M 109 ->	L 18 : 1	L 20 : 1	L 25 : 1	L 31 : 1
	L 32 : 1	L 34 : 1	L 40 : 1	L 43 : 1
	L 44 : 1			
M 110 ->	L 2 : 1	L 3 : 1	L 17 : 1	L 30 : 1
	L 36 : 1	L 40 : 1	L 43 : 1	L 44 : 1
M 111 ->	L 7 : 1	L 18 : 1	L 23 : 1	L 25 : 1
	L 34 : 1	L 40 : 1	L 43 : 1	L 44 : 1
M 112 ->	L 19 : 1	L 23 : 1	L 33 : 1	L 40 : 1
	L 43 : 1	L 44 : 1		
M 113 ->	L 19 : 1	L 25 : 1	L 31 : 1	L 32 : 1
	L 34 : 1	L 40 : 1	L 43 : 1	L 44 : 1
M 114 ->	L 2 : 1	L 3 : 1	L 26 : 1	L 31 : 1
	L 36 : 1	L 40 : 1	L 43 : 1	L 44 : 1
M 115 ->	L 18 : 1	L 20 : 1	L 23 : 1	L 25 : 1
	L 34 : 1	L 40 : 1	L 43 : 1	L 44 : 1
M 116 ->	L 7 : 1	L 18 : 1	L 23 : 1	L 26 : 1
	L 34 : 1	L 40 : 1	L 43 : 1	L 44 : 1
M 117 ->	L 19 : 1	L 23 : 1	L 25 : 1	L 34 : 1
	L 40 : 1	L 43 : 1	L 44 : 1	
M 118 ->	L 18 : 1	L 20 : 1	L 23 : 1	L 26 : 1
	L 34 : 1	L 40 : 1	L 43 : 1	L 44 : 1
M 119 ->	L 19 : 1	L 23 : 1	L 26 : 1	L 34 : 1
	L 40 : 1	L 43 : 1	L 44 : 1	