SONG SAN WOEI

ANÁLISE DE INVARIANTES NA REDE DE PETRI GHENESYS

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia.

Área de Concentração:
Engenharia Mecânica

Orientador:
Prof. Dr. José Reinaldo Silva

São Paulo
1997

Aos meus pais.

AGRADECIMENTOS

Ao Prof. Dr. José Reinaldo Silva, pela orientação deste trabalho. Ao Prof. Dr. Paulo Eigi Miyagi, pela paciência e pelas sugestões que contribuíram para este trabalho. Aos meus colegas e amigos Cristina Toshie Motohashi Matsusaki, Fabio Kawaoka Takase, Tomaz Mikio Sasaki e José Ricardo Sebastião, pelas discussões relacionadas ao tema. Finalmente, a toda minha família e a todos os meus amigos, que muito me apoiaram e incentivaram.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE SÍMBOLOS

$\varnothing$	Conjunto vazio
$\mathbf{N}$	Conjunto dos números naturais (inteiros não negativos)
∞	Infinito
$\times$	Produto cartesiano
$\cup, \cap$	União, Interseção de conjuntos
$\in$	Pertence (elemento em relação a conjunto)
$\subseteq$	Contido ou Igual (relação entre dois conjuntos)
A	Matriz de Incidência
$R(M_0)$	Conjunto de marcações atingíveis
.	Multiplicação ou produto escalar
\	Divisão Inteira
¦	Concatenação de matrizes
T	Transposta da matriz

Na definição da rede Ghenesys (capítulo 3):

::	Definição de classe
< >	Atributos
|	Concatenação
&	Agregado

RESUMO

Análise de invariantes é uma das formas de verificação de propriedades em redes de Petri. Faz parte dos chamados métodos estruturais de análise, pois depende apenas da estrutura da rede e não da sua marcação inicial. As vantagens da análise de invariantes sobre outras formas de análise são a possibilidade de se estudar o comportamento de subredes, sem a necessidade de se conhecer o comportamento da rede como um todo, e a utilização de algoritmos conhecidos de Álgebra Linear facilitando os cálculos.

A rede estendida Ghenesys é baseada na metodologia PFS/MFG com a introdução de conceitos de orientação a objetos. Entre outras modificações, a rede Ghenesys possui uma equação de estado, o que permite a sua simulação em qualquer nível de abstração e, como é mostrado neste trabalho, permite também a aplicação da análise de invariantes.

É mostrado também que, devido à forma estruturada e hierárquica da rede Ghenesys, é possível fazer-se a análise de forma estruturada também, o que é uma vantagem deste método, tanto em termos de desenvolvimento e interpretação como em termos computacionais. Finalmente, o cálculo de invariantes é aplicado num exemplo de rede Ghenesys para validar o método proposto.

ABSTRACT

Invariant Analysis is one of the methods to verify properties of one Petri net. It belongs to the group of structural methods, because it is based only on the net structure and is independent of the initial marking. The advantages of Invariant Analysis over other analyzing methods are that analysis can be performed on subnets while ignoring how the whole system behaves and the utilization of known Linear Algebra algorithms to calculate the invariants.

Ghenesys is an extended Petri net, based on the PFS/MFG methodology, with the introduction of object-oriented concepts. Among other modifications, Ghenesys has a State Equation, what enables its simulation in any abstraction level and, as it is shown in this work, also enables the application of invariant analysis.

Due to the structured and hierarchical form of Ghenesys, the invariant analysis can be done in a structured way, too, what is an advantage of this method, in the development, interpretation and computation of the invariants. Finally, the calculation of invariants is applied in a Ghenesys example to validate the proposed method.

CAPÍTULO 1 - INTRODUÇÃO

1.1 Objetivo

O objetivo deste trabalho é aplicar o método de invariantes para análise da rede de Petri Ghenesys, explorando suas propriedades de hierarquia e encapsulamento e sua complementaridade em relação às redes de Petri elementares.

1.2 Justificativa

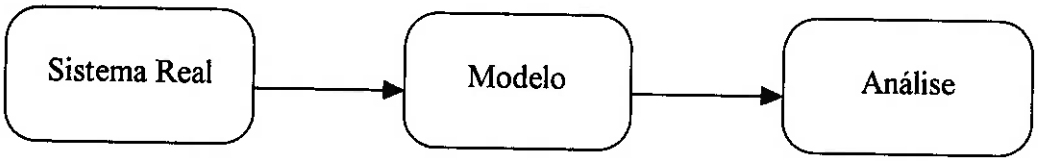


Figura 1.1 - Introdução

Tradicionalmente, os sistemas estudados em Engenharia envolvem variáveis contínuas e têm uma dinâmica que pode ser descrita através de equações diferenciais ou diferença. Recentemente, começou-se a estudar uma nova classe de sistemas, como sistemas de manufatura, de transporte, de redes de computadores e outros construídos pelo homem [Levis, 87]. Estes sistemas realizam tarefas que são caracterizadas por uma dinâmica decorrente da ocorrência de eventos que podem ser considerados de natureza discreta e assíncrona [Miyagi, 96] e assim podem ser classificados como Sistemas a Eventos Discretos (SED). Estas tarefas envolvem importantes conceitos como concorrência (compartilhamento de uma quantidade limitada de recursos), paralelismo (execução de mais de uma tarefa sem dependência entre elas) e sincronização (início ou término simultâneo de tarefas).

Existem várias ferramentas para modelagem de SEDs, como a teoria de filas, álgebra min-max, cadeias de Markov, etc. [Cao, 90] Não existe ainda um consenso sobre qual a ferramenta mais adequada [Ho, 87], mas pelas suas características gráficas e matemáticas as Redes de Petri têm sido estudadas e aplicadas em diversos casos práticos [Murata, 89] [Silva M., 89].

Uma importante característica numa ferramenta de modelagem é que ela permita analisar o modelo antes da implementação, caracterizando seu comportamento dinâmico e verificando se os requisitos de projeto do sistema estão sendo atendidos. No caso das redes de Petri os métodos de análise podem ser classificados em dois grupos:

- baseados no comportamento dinâmico, explorando o espaço de estados (configurações) que o sistema pode atingir.

- baseados em análise estrutural, ou seja, no inter-relacionamento entre os elementos da rede.

Os métodos do primeiro grupo fornecem uma quantidade maior de informações para a análise de propriedades do sistema, porém apresentam o problema de crescimento exponencial do espaço de estados (configurações), que pode ser até mesmo infinito. Existem algumas formas para se evitar a necessidade de percorrer todas as configurações possíveis (basicamente agrupando-as), mas com isso perde-se parte das informações sobre as propriedades do sistema.

As técnicas para caracterização de invariantes pertencem ao segundo grupo. Intuitivamente, um invariante é uma asserção que é válida para todos as configurações do sistema e que caracteriza as restrições de projeto ou seus requisitos básicos. As vantagens da análise de invariantes são a possibilidade de se estudar o comportamento de

subredes, sem a necessidade de se conhecer o comportamento da rede como um todo, e a utilização de algoritmos conhecidos de Álgebra Linear facilitando os cálculos. Mesmo não sendo tão conclusivo para análise do comportamento dinâmico do sistema, ainda é possível obter importantes informações sobre as propriedades do sistema.

Redes de Petri surgiram a partir da tese de doutorado apresentada por C. A. Petri em 1962 e desde então variações diversas têm sido desenvolvidas de acordo com as necessidades e enfoques adotados. Assim, existem atualmente diferentes classificações para as redes de Petri [Bernardinello, 92]. No entanto, sua utilização para a modelagem de sistemas reais apresenta algumas dificuldades. Por exemplo, à medida que o tamanho da rede aumenta, torna-se muito difícil manter uma interpretação consistente dos elementos básicos do sistema. Outro problema é a síntese da rede para sistemas complexos ou de grande porte.

Visando resolver estas dificuldades, foi desenvolvida a rede de Petri estendida Ghenesys ("*General Hierarchical Extended Net System*" [Silva J.R., 95], [Silva J.R., 96], [Shimada, 97]), baseada na metodologia PFS/MFG ([Miyagi, 88], [Miyagi, 96]), com a introdução de conceitos de orientação a objetos. A rede Ghenesys consegue combinar a metodologia *top-down* estruturada do PFS com a característica *bottom-up* do MFG [Hasegawa, 84], possuindo ainda uma equação de estado que permite a sua simulação em qualquer nível de abstração.

Os métodos de análise em redes de Petri estão baseados no formalismo matemático da rede de Petri. Assim, quando este formalismo é alterado (como para a criação da rede Ghenesys), os métodos de análise devem ser reconsiderados para este novo formalismo.

1.3 Metodologia

Este trabalho pode ser dividido nas seguintes fases:

Fase 1. Estudo do método de invariantes em redes de Petri do tipo Lugar/Transição [Murata, 89]. O estudo começa com esta classe de rede de Petri pois foi para a qual o método foi definido inicialmente. Esta fase inclui o levantamento da bibliografia existente, conceitos envolvidos e a utilização destes invariantes para análise de propriedades da rede.

Fase 2. Estudo dos algoritmos clássicos para o cálculo de soluções inteiras de um sistema de equações lineares. Estes algoritmos estão descritos tanto em trabalhos ligados à área de rede de Petri, como nas áreas de Álgebra Linear e Programação Linear.

Fase 3. Extensão do método para a rede Ghenesys. Esta fase envolve a comparação entre a rede Ghenesys e a rede Lugar/Transição, sintetizando para a rede Ghenesys um algoritmo de análise de invariantes.

Fase 4. Implementação e teste dos algoritmos, usando o software MAPLE V [Cheb-Terrab, 93]. Foi escolhido este software por ele apresentar várias rotinas prontas para manipulação de matrizes.

Fase 5. Aplicação do método de invariantes num exemplo de rede Ghenesys.

1.4 Revisão Bibliográfica

Neste item será apresentada uma breve revisão bibliográfica sobre os artigos e livros que deram suporte a este trabalho. A definição dos conceitos envolvidos com a análise de invariantes está no próximo capítulo.

[Murata, 89], [Reisig, 85] e [Peterson, 81] são textos clássicos que apresentam a teoria de Redes de Petri. [Reisig, 85] trata mais detalhadamente das redes Condição/Evento, enquanto [Murata, 89] explora as redes Lugar/Transição. A notação utilizada neste trabalho são as mais comumente utilizadas em redes de Petri e que aparece em [Murata, 89].

[Lautenbach, 87] é um texto básico sobre invariantes em redes Lugar/Transição, mostrando sua definição e aplicação na análise de propriedades.

[Schrijiver, 86] é um livro que trata sobre Álgebra Linear, Programação Linear e principalmente Programação Inteira. É discutido neste texto o problema de se achar as soluções inteiras não-negativas de um sistema de equações lineares. Outro livro sobre Álgebra Linear é [Kreyszig, 84]

[Colom, 90a], [Krückeberg, 87], [Memmi, 87] e [Treves, 89] apresentam e comparam diversos algoritmos para cálculo de invariantes. Estes algoritmos estão descritos no capítulo 2.

[Barkaoui, 92], [Colom, 90b], [Colom, 90c], [Desel, 94], [Ezpeleta, 93], [Ezpeleta, 95], [Hasegawa, 94], [Silva M., 91], [Yamalidou, 96] apresentam aplicações do método de invariantes na análise de propriedades de redes de Petri. [Silva M., 91] traz uma revisão da utilização dos invariantes para análise de propriedades. [Barkaoui, 92] e [Ezpeleta, 95] mostram que, para determinadas subclasses de redes, os invariantes fornecem condições necessárias e suficientes para determinar se uma rede é viva ou se apresenta *deadlock*. Em [Colom, 90b] e [Desel, 94], os invariantes são usados para análise de alcançabilidade. [Colom, 90c] trata da utilização de invariantes para análise de vivacidade em redes Lugar/Transição. [Ezpeleta, 93] mostra como o algoritmo para

cálculo de invariantes pode ser usado para achar sifões e *traps*. [Hasegawa, 94] usa os invariantes para fazer a transformação de uma rede C/E em MFG preservando seu comportamento dinâmico. [Yamalidou, 96] usa os invariantes não para análise, mas para síntese de uma rede de Petri de controle, de forma que o sistema controlado atenda às restrições do sistema.

[Genrich, 81] e [Jensen, 90] são textos que definem, respectivamente, o formalismo das redes de alto níveis Predicado/Transição e Colorida. Os textos mostram, junto com o novo formalismo, alterações nas técnicas de análise da rede, entre elas o novo conceito de invariantes nestas classes de redes. [Christensen, 92], [Couvreur, 90], [Couvreur, 93] são outros textos que tratam sobre cálculo e aplicações de invariantes em redes de alto nível.

[Miyagi, 88] e [Miyagi, 96] são textos que apresentam a definição do MFG e da metodologia PFS/MFG (na formulação original) e sua aplicação na modelagem e controle de sistemas de manufatura. [Silva J.R., 95] e [Silva J.R., 96] apresenta o formalismo da rede de Petri Ghenesys, na época ainda com o nome de PFS/MFG. [Shimada, 97] mostra a aplicação da rede Ghenesys num problema de planejamento.

1.5 Sumário Estruturado

O Capítulo 1 apresenta uma introdução a este trabalho, descrevendo seu objetivo e motivação.

O Capítulo 2 apresenta as definições, formalismos e algoritmos para cálculo de invariantes em redes Lugar/Transição. Apresenta também suas aplicações na análise de propriedades de uma rede.

No Capítulo 3, é apresentado o formalismo da rede Ghenesys, comparando-o com o da rede Lugar/Transição com o objetivo de sintetizar um algoritmo de análise de invariantes.

O capítulo 4 apresenta os resultados (generalização do método de invariantes para a rede Ghenesys) e mostra um exemplo de uma aplicação. Finalmente, no capítulo 5 estão as conclusões e considerações finais.

Fazem parte do trabalho ainda 4 apêndices, contendo algumas definições extras e a listagem das rotinas desenvolvidas em MAPLE.

CAPÍTULO 2 - ANÁLISE DE INVARIANTES

2.1 Definições

2.1.1 Definições Gerais

Def: uma tripla N=(P, T; F) é chamada **rede** se, e somente se, (abrev. sse):

(1) P e T são conjuntos finitos;

(2) $P \cap T = \varnothing$; $P \cup T \neq \varnothing$;

(3) $F \subseteq (P \times T) \cup (T \times P)$ é a relação de fluxo de N.

Def: Uma **marcação** M: $P \rightarrow \mathbf{N}$ numa rede N é uma função que associa um número natural a cada elemento de P.

Def: Uma rede **Lugar/Transição** (P/T) é uma 6-tupla, $R = (P, T, F, W, K, M_0)$ onde:

(i) P e T são conjuntos finitos, não-vazios, de lugares e transições

(ii) $P \cap T = \varnothing$, $P \cup T \neq \varnothing$,

(iii) $F \subseteq (P \times T) \cup (T \times P)$ é a relação de fluxo de R.

(iv) W: $F \rightarrow \mathbf{N} - \{0\}$ é uma função peso associada aos arcos,

(v) K: $P \rightarrow \mathbf{N} \cup \{\infty\}$ é uma função capacidade associada aos lugares,

(vi) M_0: $P \rightarrow \mathbf{N}$ é a marcação inicial.

Def: Para $x \in P \cup T$,

$\bullet x = \{ y \mid (y,x) \in F \}$ é chamado **pré-conjunto** de x (elementos de entrada de x).

$x\bullet = \{ y \mid (x,y) \in F \}$ é chamado **pós-conjunto** de x (elementos de saída de x).

Como relações de fluxo sempre conectam elementos de classes distintas, o pré- e pós-conjunto de um lugar são conjuntos de transições e o pré- e pós-conjunto de uma transição são conjuntos de lugares. Esta notação pode ser estendida para um conjunto de lugares ou um conjunto de transições, por exemplo, para $E \subseteq P$, $\bullet E$ é a união de $\bullet p_i$ para todos p_i pertencentes a E. Por exemplo, na figura 2.1, seja $E=\{p_1, p_3\}$, então $\bullet p_1 = \{T_1\}$, $\bullet p_3 = \{T_3\}$ e $\bullet E = \bullet p_1 \cup \bullet p_3 = \{ T_1, T_3\}$

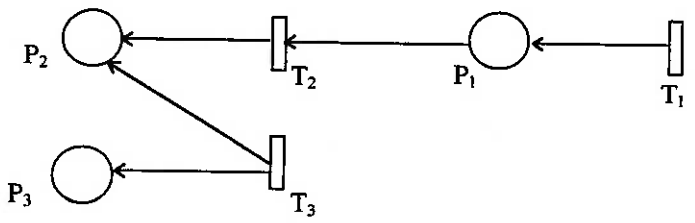


Figura 2.1 - Exemplo de rede

Def: Um elemento $x \in R$ é chamado **isolado** sse $\bullet x \cup x\bullet = \varnothing$. Normalmente, elementos isolados não são permitidos numa rede.

Def: Um par de elementos $(x,y) \in R$ é chamado um **self-loop** (fig. 2.2) sse $x \in \bullet y$ e $y \in \bullet x$. Uma rede é chamada **pura** sse não contém nenhum *self-loop*.

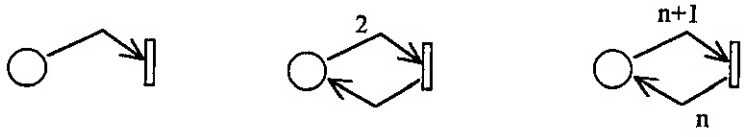


Figura 2.2 - Self-loops

Def: Uma rede P/T onde todos os arcos têm peso um é chamada **ordinária**.

Def: Uma transição t está **habilitada** numa marcação M se cada lugar $p \in {}^\bullet t$ estiver marcado com pelo menos w(p, t) marcas (onde w(p, t) é o peso do arco que vai de p para t) e o número de marcas em cada lugar $p \in t^\bullet$ for menor ou igual a K(p) - w(t,p) (onde w(t, p) é o peso do arco que vai de t para p e K(p) é a capacidade do lugar p). Em outras palavras, uma transição está habilitada se cada lugar pertencente a seu pré-conjunto tiver marcas suficientes e houver espaço suficiente para marcas nos lugares pertencentes ao seu pós-conjunto.

Numa rede onde todos os lugares tiverem capacidade infinita, sempre haverá espaço suficiente nos lugares de saída. Desta maneira, para descobrir quais transições estão habilitadas, basta verificar o número de marcas nos lugares de entrada de cada transição. Neste capítulo, por simplificação, serão consideradas apenas redes onde **todos** os lugares têm capacidade infinita. No próximo capítulo será analisado o caso de uma rede onde os lugares têm capacidade finita.

Uma transição habilitada pode ou não disparar. O disparo de uma transição habilitada retira w(p, t) marcas de cada lugar $p \in {}^\bullet t$ e acrescenta w(t, p) marcas em cada lugar $p \in t^\bullet$.

Def.: Para uma rede P/T **pura** contendo n transições e m lugares, sua **Matriz de Incidência** $A = [a_{ij}]$ é uma matriz de inteiros n x m cujos elementos são dados por

$$a_{ij} = a_{ij}^{+} - a_{ij}^{-}$$

onde a_{ij}^{+} é o peso do arco que vai da transição i para o lugar j (ou zero se não houver arco da transição i para o lugar j) e a_{ij}^{-} é o peso do arco que vai do lugar j para a transição i (zero, se não houver arco do lugar j para a transição i).

Para utilizar esta definição para a Matriz de Incidência, é necessário que a rede seja pura, senão redes de comportamentos diferentes, como na figura 2.2, teriam a mesma representação matricial, tornando insatisfatória a análise baseada na matriz de incidência.

Para representar redes não puras, é necessário definir 2 matrizes, como em [Ezpeleta, 93], onde as matrizes são chamadas de matriz de pré-incidência e matriz de pós-incidência.

Representando cada marcação (estado) como um vetor m x 1, a mudança de estado pode ser representada numa equação (a **Equação de Estado**), dada por

$$M_k = M_{k-1} + A^T u_k \qquad k = 1, 2, \ldots$$

onde o vetor de controle u_k é um vetor n x 1 formado por zeros e um único 1 na linha i, indicando o disparo da transição i.

Escrevendo a Equação de Estado para k = 1, 2, ..., n e somando-as, temos:

$$M_n = M_0 + A^T \sigma$$

onde $\sigma = \sum_{k=1}^{n} u_k$ é chamado de vetor de contagem de disparos.

Def: Uma marcação M_k é **atingível** (alcançável) a partir de M_0 se existir uma seqüência de vetores $u_1, u_2, \ldots, u_k$ que levem de M_0 para M_k segundo a Equação de Estado. O conjunto de marcações atingíveis a partir de M_0 é representado por $R(M_0)$.

Def: Uma marcação M_k é **coberta** se existir uma marcação M' em $R(M_0)$ tal que $M'(p) \geq M_k(p)$ para todos os lugares p da rede.

2.1.2 Propriedades

Podem ser definidos dois tipos de propriedades para redes de Petri [Murata, 89]:

Propriedades Comportamentais

As propriedades comportamentais são aquelas que dependem da marcação inicial da rede:

i) Limitação (*Boundedness*)

Uma rede P/T é k-limitada (sendo k um número inteiro positivo) se o número de marcas em cada lugar não exceder o valor k para qualquer marcação M_k em $R(M_0)$. Particularmente, uma rede é segura (*safe*) se for 1-limitada.

ii) Vivacidade (*Liveness*)

Uma rede é viva se, para qualquer marcação M_k em $R(M_0)$, qualquer transição possa ser disparada em alguma seqüência de disparos. Um conceito relacionado a este é o de *deadlock* , isto é, uma marcação do sistema onde nenhuma transição possa mais disparar.

iii) Reversibilidade e *Home State*

Uma rede é reversível se, para qualquer marcação M_k em $R(M_o)$, M_o é alcançável a partir de M_k. Ou seja, qualquer que seja o estado do sistema, é sempre possível voltar ao estado inicial.

Em algumas aplicações, a condição de reversibilidade é relaxada, sendo suficiente voltar a algum estado M' (chamado de *home state*).

Propriedades Estruturais

As propriedades estruturais são aquelas que dependem da estrutura da rede, ou seja, como os elementos da rede estão relacionados. Estas propriedades são chamadas independentes da marcação inicial porque são válidas para qualquer marcação inicial ou porque estão relacionadas à existência de seqüências de disparos para alguma marcação inicial. Estas propriedades podem ser caracterizadas em função da matriz de incidência A de uma rede P/T pura.

i) Vivacidade Estrutural

Uma rede é viva estruturalmente se ela for viva para alguma marcação inicial finita. Ou seja, uma condição necessária (mas não suficiente) para uma rede ser viva é que ela seja estruturalmente viva.

ii) Controlabilidade

Uma rede é completamente controlável se qualquer marcação for alcançável a partir de qualquer outra marcação. Se uma rede de Petri com m lugares for completamente controlável, então o posto da sua matriz de incidência é igual a m (para que a equação de estado tenha sempre solução) [Murata, 89].

iii) Limitação Estrutural

Uma rede é estruturalmente limitada se ela for limitada para qualquer marcação inicial finita. Diferente da vivacidade estrutural, limitação estrutural é uma condição suficiente (mas não necessária) para limitação. Considerando a matriz de incidência A, a condição necessária e suficiente para uma rede ser estruturalmente limitada é existir um vetor $x_{m \times 1}$ de componentes inteiros positivos tal que $A.x \leq 0$ [Murata, 89].

iv) Conservação

Uma rede é conservativa se existir um vetor $x_{m \times 1}$ de componentes inteiros positivos tal que a soma ponderada de marcas seja constante ($M^T x = M_0^T x = const.$) para uma marcação inicial M_0 qualquer e para todo M em $R(M_0)$. Considerando a matriz de incidência A, a condição necessária e suficiente para uma rede ser conservativa é existir um vetor $x_{m \times 1}$ de componentes inteiros positivos tal que $A.x = 0$ [Murata, 89].

v) Repetitividade

Uma rede é repetitiva se existir uma marcação inicial M_0 e uma seqüência de disparos σ a partir de M_0 tal que toda transição ocorra infinitas vezes em σ. Considerando a matriz de incidência A, a condição necessária e suficiente para uma rede ser repetitiva é existir um vetor $y_{n \times 1}$ de componentes inteiros positivos tal que $A^T . y \geq 0$ [Murata, 89].

vi) Consistência

Uma rede é consistente se existir uma marcação inicial M_0 e uma seqüência de disparos σ a partir de M_0 que leva o sistema de volta a M_0 tal que toda transição ocorra pelo menos uma vez em σ. Considerando a matriz de incidência A, a condição necessária e suficiente para uma rede ser consistente é existir um vetor $y_{n \times 1}$ de componentes inteiros positivos tal que $A^T . y = 0$ [Murata, 89].

2.1.3 Elementos Estruturais

Def: Dada uma rede R e uma marcação inicial M_0 , *p* é um **lugar implícito (IP)** se, removendo-o, o conjunto de marcações atingíveis não se alterar.

Def: Dada uma rede R, *p* é um **lugar estruturalmente implícito (SIP)** se, qualquer que seja a marcação inicial da rede, existir uma marcação inicial finita para *p* que torna *p* um lugar implícito.

Def: Seja $E \subseteq P$ um conjunto de lugares de uma rede R.

(a) E é um **sifão** se e somente se $\bullet E \subseteq E\bullet$

(b) E é um **trap** se e somente se $E\bullet \subseteq \bullet E$.

(c) E é um **sifão-trap** se e somente se $E\bullet = \bullet E$.

Traps são conjuntos de lugares que, uma vez marcados, permanecem marcados, enquanto sifões são conjuntos de lugares que, uma vez com marcação zero, permanecem sem marcas. Estes elementos são importantes para análise da vivacidade da rede, pois se um sifão ficar vazio, suas transições de saída não ficam mais habilitadas.

Def: Um vetor $y_{n \times 1}$ com componentes inteiros, solução da equação homogênea $A^T \cdot y = 0$ é um **T-invariante** ou *T-flow*.

Def: Um vetor $x_{m \times 1}$ com componentes inteiros, solução da equação homogênea $A \cdot x = 0$ é um **S-invariante** ou *S-flow*.

Def: Um invariante com componentes inteiros não-negativos, é chamado de ***semiflow*** ***(S-semiflow*** ou *T-semiflow)*.

Def: Para um *semiflow* $w = (w_1, w_2, ...)$, o conjunto $\| w \| = \{i \mid w_i > 0\}$ é o **suporte** de w. Em outras palavras, o suporte de um *semiflow* w é o conjunto dos índices que correspondem a componentes de w maiores que zero.

Def: O suporte de um *semiflow* w é **mínimo** sse não contém estritamente o suporte de nenhum outro *semiflow*.

Def: Um *semiflow* w é **canônico** sse o máximo divisor comum de seus componentes é igual a um.

Def: Um *semiflow* w é **mínimo** sse ele é canônico e seu suporte é mínimo.

Por exemplo, suponha que uma rede possua como invariantes, entre outros, os vetores $x_1 = (1\ 1\ 2\ 2\ 0)^T$ e $x_2 = (2\ 0\ 1\ 0\ 0)^T$. Assim:

- x_1 e x_2 são canônicos
- $||x_1|| = \{1, 2, 3, 4\}$ e $||x_2|| = \{1, 3\}$
- x_2 é mínimo, mas x_1 não é.

2.1.3 Propriedade dos T-Invariantes

T-Invariantes estão associados com seqüências de disparos que fazem o sistema retornar a uma marcação dada. Numa rede P/T onde todos os lugares tiverem capacidade infinita, vale o seguinte teorema:

Teorema: Um vetor $y \geq 0$ é um T-invariante se e somente se existir uma marcação M e uma seqüência de disparos $u_1, u_2, \ldots, u_k$ tal que $y = \Sigma\, u_i$ leva o sistema de volta a M.

Dem.:

($\Leftarrow$): Seja $y \geq 0$ uma seqüência de disparos, que leva o sistema de uma marcação M de volta a essa mesma marcação. Então, pela equação de Estado:

$$M = M + A^T y \Rightarrow A^T y = 0$$

($\Rightarrow$)[Reisig, 85]: Seja $y \geq 0$ um T-invariante, ou seja, $A^T y = 0$. O suporte de y corresponde a um conjunto de transições. Seja t_1 uma destas transições. Primeiramente, construímos uma marcação M_1 onde todos os lugares pertencentes a $\bullet t_1$ estão marcados. Como todos os lugares têm capacidade infinita, então t_1 está habilitada nesta marcação. Repetindo este procedimento para as demais transições correspondentes ao suporte de y e somando todas as marcações, obtemos uma marcação M dada por $M = M_1 + M_2 + ...$ Como os lugares têm capacidade infinita, esta marcação é possível, e nesta marcação todas as transições correspondentes ao suporte de y estão habilitadas.

Após os disparos de y, a nova marcação obtida será

$$M_k = M + A^T y = M + 0 \Rightarrow M_k = M \quad \square$$

Se a rede considerada tiver capacidade finita nos lugares, então este teorema não é mais verdadeiro e precisa ser alterado: pode existir um T-invariante y que não corresponde a nenhuma seqüência de disparos possível. A recíproca, no entanto, continua válida: se existir uma seqüência de disparos que leva o sistema de volta a uma determinada marcação M, então o vetor correspondente será um T-invariante.

2.1.4 Propriedade dos S-Invariantes

S-Invariantes estão associados com a propriedade de conservação da soma ponderada de marcas num dado conjunto de lugares, em todas as marcações atingíveis.

Teorema: um vetor x é um S-invariante sse

$$M^T . x = M_0^T . x$$

para qualquer marcação inicial M_0 e qualquer marcação M atingível a partir de M_0.

Dem[Reisig, 85]:

($\Rightarrow$): Suponha $x \geq 0$, com $A \cdot x = 0$.

Transpondo a Equação de Estado da rede e multiplicando por x, temos:

$\Delta M^T \cdot x = (\sigma^T A) \cdot x = \sigma^T (A \cdot x) = \sigma^T .0 = 0$

Então $(M - M_0)^T \cdot x = 0 \Rightarrow M^T.x = M_0{}^T.x$

($\Leftarrow$): Suponha $M^T.x = M_0{}^T.x$ para todo M atingível.

$\Delta M^T \cdot x = (M - M_0)^T \cdot x = 0$

Então $(\sigma^T A) \cdot x = \sigma^T (A \cdot x) = 0$

Esta igualdade deve ser verdadeira para todo vetor de contagem de disparos σ, portanto $A \cdot x = 0$. □

Uma conseqüência desta propriedade de conservação de marcas é o limite máximo de marcas que um lugar *p* pode conter [Murata89]. Por exemplo, seja x um S-*semiflow*. A propriedade de conservação da soma ponderada de marcas mostra que

$M^T. x = M_0{}^T. x$

Como o número de marcas em cada lugar não pode ser negativo e, por hipótese, os componentes de x também não são negativos, então o termo à esquerda da igualdade corresponde a uma soma de números inteiros não-negativos. Cada parcela da soma, portanto, é menor ou igual ao total dado pelo termo à direita da igualdade. Se o componente de x correspondente ao lugar *p* (chamado de x_p) for maior que zero, temos:

$$M(p)\, x_p \leq M_0^T .\, x \Rightarrow M(p) \leq (M_0^T .\, x) / x_p .$$

Este valor máximo para o número de marcas no lugar *p* precisaria ser calculado para todos os S-*semiflows*, mas [Murata, 89] mostra que o menor valor sempre ocorre com um S-*semiflow* mínimo. Isto é:

$$M(p) \leq Mín[(M_0^T .\, x) / x_p]$$

onde o mínimo é tomado sobre todos os S-semiflows mínimos com $x_p > 0$

2.2 Cálculo de Invariantes

Existem várias técnicas para o cálculo das soluções de uma equação matricial homogênea C.x = 0, onde C = A (matriz de incidência) no caso de S-Invariantes e C = A^T (matriz de incidência transposta) no caso de T-Invariantes. Da álgebra linear, sabe-se que as soluções desta equação formam um espaço vetorial, também chamado de espaço anulador de C [Kreyszig, 84], ortogonal ao espaço formado pelas linhas de C. Uma base para este espaço vetorial pode ser encontrada usando, por exemplo, o método de triangularização de Gauss através de operações elementares sobre as linhas da matriz C (soma de duas linhas, multiplicação de uma linha por escalar). A base encontrada por este método, no entanto, poderá conter números fracionários ou negativos.

Para análise da rede, entretanto, é importante obter os invariantes de componentes inteiros não-negativos (os *semiflows*), pois são os que tem interpretação sobre a rede. Por exemplo, cada componente de um T-invariante corresponde ao disparo de uma transição; não tem significado físico um número negativo de disparos.

2.2.1 Invariantes de componentes inteiros

[Treves, 89] apresenta um algoritmo para encontrar uma base para o espaço anulador de C, contendo apenas componentes inteiros:

Inicialização: $Q_0 = I_m$, $A_0 = C$, $T_0 = \varnothing$, onde m e n são o número de lugares e transições na rede, respectivamente, e I_m é a matriz identidade de ordem m

Para i = 1 até m
Repetir
card é o número de elementos não nulos da linha i de A_{i-1}
q é o índice do 1º elemento de menor valor absoluto (não-nulo) da linha i de A_{i-1}
se card = 1 então
$T_i = T_{i-1} \cup \{q\}$, $[A_i \mid Q_i] = [A_{i-1} \mid Q_{i-1}]$
senão
para k=1 até n
para p=1 até m, $p \neq q$
$[A_{i-1}(k,p) \mid Q_{i-1}(k,p)] = [A_{i-1}(k,p) \mid Q_{i-1}(k,p)] - A_{i-1}(i,p) \setminus A_{i-1}(i,q) * [A_{i-1}(k,p) \mid Q_{i-1}(k,p)]$
até que card ≤ 1

Após o passo m, A_m é chamado forma normal de Hermite (ver apêndice) da matriz A e as colunas de $Q_m' = (Q_m(i,j)$ tal que $j \notin T_m)$ formam uma base de invariantes de componentes inteiros. Em [Schrijiver, 86] é mostrado que este algoritmo possui complexidade polinomial.

Observe que este algoritmo é similar ao método de triangularização de Gauss, só que utiliza divisões inteiras sucessivas para evitar o aparecimento de números fracionários. Por exemplo, suponha que numa determinada linha os únicos elementos não nulos sejam 5 e 3. As sucessivas iterações seriam:

$$[0\ 5\ 0\ 3] \Rightarrow [0\ 2\ 0\ 3] \Rightarrow [0\ 2\ 0\ 1] \Rightarrow [0\ 0\ 0\ 1]$$

2.2.2 Semiflows

O conjunto de *semiflows* pode ser infinito, pois se w_1 e w_2 forem *semiflows*, $w_1 + w_2$ também será. Apesar de não formarem um espaço vetorial, é possível encontrar um

gerador de *semiflows*, $W=\{w_1, w_2, ..., w_q\}$, isto é, um conjunto mínimo de *semiflows* com a propriedade de que qualquer *semiflow* w pode ser obtido da combinação linear dos elementos deste conjunto. Qualquer *semiflow* w pode então ser representado por

$w = \Sigma (k_j \ w_j)$, onde $w_j \in W$ e k_j é um número racional.

[Colom, 90a], [Treves, 89] mostram que, para uma rede P/T, este conjunto W (o gerador dos *semiflows*) é finito e único, e os elementos de W são os *semiflows* mínimos (definidos em 2.1.3).

Existem duas abordagens para a obtenção dos *semiflows* mínimos de uma rede: na primeira, encontra-se os *semiflows* diretamente a partir da matriz de incidência da rede e na segunda é encontrada primeiro uma base de invariantes de componentes inteiros, para, a partir deles, obter os *semiflows* mínimos. Por simplificação, os algoritmos apresentados são para cálculo de S-semiflow. Para o cálculo de T-semiflows, basta transpor a matriz de incidência.

Algoritmo 1 - Obtendo os semiflows mínimos a partir da matriz de incidência A

(1) C := A, Y := Im { Im é a matriz identidade de ordem m (nº de lugares na rede) }
(2) Para i := 1 até n { n é o número de transições na rede }
2.1 Adicione à matriz [Y ¦ C] todas as linhas que são combinações lineares de pares de linhas de [Y ¦ C] e que anulam a i-ésima coluna de C.
2.2 Elimine de [Y ¦ C] as linhas em que a i-ésima coluna de C não é nula.
(3) As linhas da matriz Y são os semiflows da rede. Entre eles estão todos os semiflows mínimos.

Algoritmo 2 - Obtendo os semiflows mínimos a partir de uma base de invariantes

(1) $C := B^T$ { B tem dimensão n x (n-r) e é uma base de invariantes }
(2) Para i := 1 até r
2.1 Some à matriz C todas as linhas resultantes de combinações lineares positivas de pares de linhas de C e que anulam a i-ésima coluna de C.
2.2 Elimine de C todas as linhas nas quais a i-ésima coluna é negativa.
(3) As linhas da matriz C são semiflow da rede. Entre eles estão todos os semiflows mínimos.

Estas duas abordagens são baseadas num único método, o algoritmo de eliminação de Fourier-Motzkin (ver apêndice), também chamado de algoritmo de Farkas em [Memmi, 87] e [Treves, 89]. Um dos problemas com estes algoritmos é com relação a tempos de execução e ocupação de memória, que crescem exponencialmente com o tamanho da matriz, devido à adição de novas linhas e devido à natureza combinatória do problema. Para evitar este crescimento exponencial, algumas melhorias podem ser adicionadas para detectar e eliminar redundâncias, junto com uma série de regras heurísticas para controlar esta explosão combinatória.

Um ponto importante para a eficiência do algoritmo 2 é a escolha da base de invariantes sobre a qual será aplicado o algoritmo. Poderia ser utilizado o método de triangularização de Gauss para a obtenção desta base, mas isto pode levar a grandes tempos de execução e ocupação de memória durante o cálculo de *semiflows*. A eficiência do algoritmo 2 depende muito da base escolhida, particularmente que a matriz contendo a base de invariantes seja esparsa (a maioria de seus termos seja igual a zero).

Os resultados obtidos em [Colom, 90a] mostraram que os dois algoritmos possuem tempos de execução da mesma ordem de grandeza, não sendo possível determinar a priori qual o melhor método para cada caso de rede.

Melhorias no Algoritmo [Colom90a]

1) Eliminação de *semiflows* não mínimos, através da comparação dos suportes.

Os algoritmos acima geram também *semiflows* que não são mínimos. Essas linhas são redundantes, assim é interessante eliminá-las para evitar o crescimento exponencial no número de linhas. Pela definição, um *semiflow* é não mínimo se, e somente se, seu

suporte contém estritamente o suporte de outro *semiflow* Assim, uma forma para se descobrir se um *semiflow* é ou não mínimo é através da comparação dos suportes. A comparação pode ser feita quando uma nova linha for gerada: se ele contiver o suporte de um *semiflow* já existente, então a combinação não será feita.

2) Utilização de uma heurística para seleção da coluna a ser anulada.

Uma heurística para seleção da coluna a ser anulada pode ser acrescentada ao algoritmo com o objetivo de limitar o número de linhas adicionadas em cada iteração. Seja P_k e N_k o número de elementos positivos e negativos, respectivamente da coluna k de A que se quer anular. O número de linhas adicionadas, combinando-se pares de linhas, é P_k . N_k. O número de linhas eliminadas no final da iteração é dado por $P_k + N_k$. O *fator de expansão* F(k) pode ser definido, então, como o número de linhas adicionadas no processo de anulação da coluna k, e é dado por:

$$F(k) = P_k \cdot N_k - (P_k + N_k)$$

A heurística é então escolher, como coluna a ser anulada, a primeira coluna com fator de expansão negativo ou, se não houver nenhuma coluna, aquela com menor valor para P_k . N_k. Entretanto, pode ser que esta escolha não seja ótima para o algoritmo como um todo.

3) Ao anular a coluna k, separação das linhas de A em 3 conjuntos, conforme o elemento na coluna k for positivo, zero ou negativo.

Seja U_0, U_+ e U_- o conjunto das linhas de A em que a coluna k for nula, positiva ou negativa. Ao anular a coluna k de A, cada linha de U_+ é combinada com cada linha em U_- e as novas linhas são adicionadas a U_0 para serem usadas no passo seguinte. Ao final deste processo, as linhas em U_+ e U_- são removidas.

O objetivo desta técnica é reduzir o número de comparações necessárias para escolher duas linhas que se anulam.

4) Teste rápido de minimalidade do *semiflow*, comparando a cardinalidade de seu suporte com um limite máximo.

Assim como em 2), esta variante elimina os *semiflows* não mínimos, evitando crescimento exponencial no número de linhas. Dessa vez, utiliza-se a seguinte propriedade dos *semiflows* mínimos: um *semiflow* x é não mínimo se a cardinalidade de seu suporte é maior que o posto da submatriz formada pelas linhas de A correspondentes aos lugares que pertencem ao suporte de x [Colom, 90a].

Na prática, para evitar calcular o posto das submatrizes, utiliza-se um limite superior para o posto, conseguindo-se assim apenas uma condição suficiente para um *semiflow* ser não mínimo. O limite superior é o número de colunas não nulas da submatriz composta pelas linhas de A correspondentes aos lugares que pertencem ao suporte do *semiflow*.

Com estas melhorias, o algoritmo fica:

Anule todas as colunas k de U_0 em que $P_k=N_k=1$
Selecione a primeira coluna não nula de U_0 tal que $P_k \cdot N_k - (P_k + N_k) < 0$ ou uma coluna k que minimize $P_k \cdot N_k$.
    Enquanto k ≠ null
        Mova as linhas j de U_{k-1} tal que $A_{k-1}[j,k] = 0$ para U_k
        Mova as linhas j de U_{k-1} tal que $A_{k-1}[j,k] < 0$ para U_-
        Mova as linhas j de U_{k-1} tal que $A_{k-1}[j,k] > 0$ para U_+
        Linha1 := primeira_linha_ U_-
        Enquanto Linha1≠ null
            Linha2 := primeira_linha_ U_+
            Enquanto Linha2≠ null
                Calcule o suporte do novo semiflow gerado na união de Linha1 e Linha2 (W)
                Se card(||W||) ≤ limite_superior_posto + 1
                    Se ||W|| não contém o suporte de um outro *semiflow*
                        Gere e adicione a nova linha a U_k
                        Divida pelo maior elemento (em módulo) desta nova linha

Guarde ||W||
Fim Se
Fim Se
Próximo Linha2
Fim Enquanto
Próximo Linha1
Fim Enquanto
Elimine as linhas U_+ e U-
Selecione a primeira coluna não nula de U_k tal que $P_k \cdot N_k - (P_k + N_k) < 0$ ou uma coluna k que minimize $P_k \cdot N_k$
Fim Enquanto

2.3 Aplicação de Invariantes

2.3.1 Propriedades Estruturais

A partir das definições das propriedades apresentadas em 2.1.2, conclui-se que:

1. Se existir um S-invariante x de componentes inteiros positivos, então a rede de Petri é conservativa e estruturalmente limitada.

2. Se existir um T-invariante y de componentes inteiros positivos então a rede de Petri é consistente e repetitiva.

2.3.2 Vivacidade

Numa rede onde todos os lugares possuem capacidade infinita, para verificar se uma transição está habilitada só é preciso verificar seus lugares de entrada. Utilizando os invariantes da rede, obtemos uma condição necessária para uma transição t poder ser habilitada [Colom, 90c]:

$$I^T \cdot M_0 \geq I^T \cdot \text{Pré}(t) \quad \text{para todo S-invariante I,}$$

onde Pré(t) é a marcação mínima necessária para que a transição t esteja habilitada, isto é, a marcação onde cada lugar de entrada da transição t está com um número de marcas igual ao peso do arco que vai deste lugar até a transição t.

Uma outra abordagem utiliza o conceito de sifão. [Ezpeleta, 93] mostra que existe um conjunto de sifões com a propriedade de que qualquer outro sifão da rede pode ser obtido da união de sifões deste conjunto. Utilizando uma terminologia semelhante à utilizada com os invariantes, este conjunto é chamado de gerador de sifões e seus elementos são os sifões mínimos. Uma forma de se encontrar estes sifões mínimos é utilizar o algoritmo para cálculo de invariantes numa rede acrescida de Lugares Implícitos. O suporte destes invariantes, excluindo os componentes correspondentes aos Lugares Implícitos, fornecem os sifões mínimos. A partir disso, uma condição necessária para que a rede seja viva é que todo sifão mínimo esteja marcado na marcação inicial.

2.3.3 Alcançabilidade

Foi mostrado no item 2.1.4 que os S-Invariantes possuem a propriedade de conservação da soma ponderada de marcas em todas as marcações atingíveis. Temos, assim, uma condição necessária (mas não suficiente) para uma determinada marcação M ser alcançável a partir de M_0: a marcação não será alcançável se existir algum S-invariante I tal que $I^T . M_0 \neq I^T . M$.

[Desel, 94] mostra que a existência de um invariante satisfazendo a condição acima é equivalente à afirmação de que a equação de estado $A^T . x = M - M_0$ não possui solução racional. Neste mesmo artigo, é introduzido o conceito de Módulo-S-Invariante, que gera uma restrição equivalente à afirmação de que a equação de estado não possui solução inteira.

[Colom, 90b] introduz uma outra técnica para melhorar esta restrição. Primeiro, é utilizada Programação Linear para incluir lugares implícitos na rede (pela definição,

que não alteram o conjunto de marcações atingíveis). Os invariantes desta rede aumentada são então encontrados e utilizados como anteriormente.

2.4 Limitações

[Lakos, 93] identifica algumas limitações na análise através de invariantes. Uma delas é que nem toda asserção pode ser representada por um S-invariante, pois o conceito de conservação de marcas é útil apenas para sistemas cíclicos. Num sistema de comunicação, por exemplo, onde as marcas são transmitidas de um emissor para um receptor, não é possível achar uma soma ponderada de marcas que se conserva, não existindo então nenhum S-invariante que caracterize as restrições de projeto. Não é o caso dos sistemas flexíveis de manufatura, normalmente compostos de estações de trabalho, onde os produtos são processados, e de um sistema de transporte para carregar e descarregar as estações. A rede que modela este tipo de sistema deve ser repetitiva, existindo invariantes que traduzem as restrições ou requisitos do sistema.

Outro problema é que a abordagem por invariantes somente fornece condições necessárias ou suficientes (não ambas ao mesmo tempo) para uma determinada propriedade. Isso acontece porque nem toda solução da equação de estado corresponde a um estado atingível pelo sistema, o que acaba não permitindo que se conclua que a rede possui determinada propriedade. Por exemplo, um S-invariante indica que a soma ponderada de marcas se conserva com o disparo das transições, mas nada garante que as transições *irão* disparar. Somente em algumas subclasses de redes Lugar/Transição ([Barkaoui, 92], [Ezpeleta, 95]), é que os invariantes fornecem condições necessárias e suficientes para análise de propriedades da rede.

CAPÍTULO 3 - REDE DE PETRI GHENESYS

3.1 Introdução

No capítulo anterior, foi apresentada a teoria e aplicação da análise de invariantes para a rede de Petri Lugar/Transição. Este tipo de rede se aplica à modelagem e análise de vários tipos de sistemas, porém sua utilização em sistemas reais apresenta algumas dificuldades. Por exemplo, à medida que o tamanho da rede aumenta, torna-se muito difícil manter uma interpretação consistente dos elementos básicos do sistema. Uma solução possível para este problema está em se utilizar redes de Petri mais sintéticas, baseadas nas redes de alto nível ou em redes estendidas como o MFG [Hasegawa, 84], dirigido para a modelagem e controle de sistemas de manufatura.

Outro problema é a síntese da rede para sistemas complexos ou de grande porte. Visando superar esta dificuldade, foi desenvolvido o PFS/MFG [Miyagi, 88], uma metodologia *top-down* para se fazer sistematicamente a modelagem e o mapeamento dos elementos físicos do sistema produtivo (representado através do PFS - *Production Flow Schema*) nos elementos da rede de Petri do tipo MFG. O produto final da aplicação da metodologia PFS/MFG é uma rede MFG, com diversos níveis de abstração. O comportamento dinâmico do modelo somente poderia ser verificada quando esta forma MFG fosse atingida.

A rede MFG (*Mark Flow Graph)* foi desenvolvida com base na experiência prática de projetistas de sistemas de manufatura. No entanto, a ausência de um formalismo matemático para esta rede dificulta a análise, principalmente no caso de sistemas grandes ou complexos (como é o caso dos sistemas flexíveis e integrados de manufatura).

Uma propriedade importante da rede MFG é a identificação de operações características (e abstratas) de chão de fábrica que são associadas a elementos especiais da representação gráfica (boxes de montagem, distribuidores, buffers, etc.). Esta mesma propriedade, entretanto, torna-se um problema em relação à análise de propriedades uma vez que invalida a representação algébrica da rede através da Matriz de Incidência e Equação de Estado como estão definidas na rede Lugar/Transição.

A rede estendida Ghenesys ("*General Hierarchical Extended Net System*") [Silva J.R., 95] é baseada no PFS/MFG com a introdução de conceitos de orientação a objetos que permite o encapsulamento de operadores em objetos passivos (da classe box) e o encapsulamento de processos nos objetos ativos (da classe atividade). A rede Ghenesys consegue assim combinar a metodologia *top-down* estruturada do PFS e ser ao mesmo tempo uma rede estendida como o MFG, com a vantagem de poder ser simulada (utilizando sua equação de estado) em qualquer nível de abstração.

Este capítulo apresenta brevemente o formalismo da rede Ghenesys, comparando-o com o da rede Lugar/Transição com o objetivo de sintetizar para a rede Ghenesys um algoritmo de análise de invariantes.

3.2 Elementos Básicos da Rede Ghenesys

A rede estendida Ghenesys [Silva J.R., 96] é um grafo bipartido, orientado a objetos, definida como uma tripla (B, A, G), onde B e A são conjuntos disjuntos de objetos chamados de boxes e atividades, respectivamente, e G contém as relações de fluxo entre os elementos pertencentes a $(A \times B) \cup (B \times A)$. Cada elemento em B ou A é representado por um objeto.

O Ghenesys possui duas super-classes principais: a classe box (elemento passivo/estático) e a classe atividade (elemento ativo/dinâmico). Cada objeto estático da classe box pode representar uma única condição ou pode representar um elemento estático composto. Métodos associados aos objetos estáticos podem representar modos específicos de armazenamento de itens (como FIFO, LIFO, etc.) ou processos de montagem e desmontagem. Da mesma forma, cada objeto dinâmico da classe atividade pode representar um único evento instantâneo ou um elemento dinâmico composto, isto é, uma subrede dinâmica.

Outro elemento existente no Ghenesys é a porta (*gate*), um tipo especial de relação entre elementos estáticos e dinâmicos da rede originados a partir de sinais externos ao sistema, como por exemplo botões liga/desliga, chave reset, etc. Um objeto com uma marca persistente (ou marca permanente) denota uma pseudo-condição (pc_i) associada à porta. As portas podem ser inibidoras ou habilitadoras, dependendo do papel exercido pela marcação persistente de inibir ou habilitar uma transição.

3.2.1 Super-Classe Box

A super-classe box é caracterizada por um conjunto mínimo de atributos box::<#nome,#marca>, onde o atributo #nome é uma variável que identifica de uma forma única o objeto box e #marca é uma variável inteira que indica o número de marcas no interior do box.

A classe box possui as seguintes subclasses: {box_temporizado, box_capacidade}. A classe Box_temporizado é caracterizado por Box_temporizado :: < box | #tempo_estimado>, isto é, ela herda todos os atributos de box e tem ainda um parâmetro extra contendo o tempo estimado para habilitar a atividade. Similarmente, a

classe box_capacidade é caracterizada por box_capacidade :: < box | #capacidade>, onde #capacidade é o número máximo de marcas que este objeto pode aceitar.

Existem ainda subclasses para a classe box_capacidade, definidas como: subrede_estática :: < box_capacidade | #relação_es, #pt_subrede>, ou seja ela herda todos os atributos da classe box-capacidade e possui ainda os atributos #relação_es, que indica uma relação de entrada-saída, e #pt_subrede que é um ponteiro para a subrede estática correspondente . Um box desmontagem, por exemplo, seria um objeto da classe subrede_estática com #relação_es = (1:n) e um box_montagem teria #relação_es = (n:1). O atributo #pt_subrede é um ponteiro para uma subrede estática.

3.2.2 A Super-Classe Atividade

A super-classe atividade é caracterizada por um conjunto mínimo de atributos atividade :: <#nome> onde o atributo #nome é uma variável que identifica de forma única o objeto atividade. Ela possui duas subclasses

- atividade_simples :: <#nome,#lista_paralelas>
- atividade_composta :: <atividade | & [2,atividade_simples], [1, box_temporizado], #pt_subrede>

Assim, a classe atividade_simples herda todos os atributos da classe atividade e possui mais o atributo #lista_paralelas com a lista de todas as outras atividades que podem ser disparadas em paralelo com ela. Desta forma, uma atividade_simples é uma definição alternativa de um evento, com a possibilidade de especificar uma lista de outras atividades que podem ser disparadas ao mesmo tempo.

A classe atividade_composta, ou subrede dinâmica, herda todos os atributos da classe atividade e, além disso, é composta por 2 atividades_simples (atividade de entrada

e saída), um box_temporizado que guarda o tempo estimado de processamento e um ponteiro #pt_subrede para outra subrede (dinâmica) que representa o modelo do processo especificado. Todos os atributos são opcionais, permitindo abstração e o uso de uma abordagem top-down.

3.2.3 As Relações de Fluxo

Cada elemento em G é uma relação definida no conjunto $(B \times A) \cup (A \times B)$ e não definem uma nova classe de objetos. Os elementos de G são classificados em fluxo normal ou porta (*gate*), no caso de apresentar marca persistente. As portas podem ser classificadas como inibidoras ou habilitadoras, dependendo do papel exercido pela marcação persistente de inibir ou de habilitar uma atividade da rede.

3.3 Diferenças para a Análise de Invariantes

A seguir, serão analisadas as diferenças entre o formalismo da rede Ghenesys e da rede Lugar/Transição, e como elas influem na análise de invariantes.

3.3.1 Habilitação das Atividades

No capítulo anterior, foram considerados apenas redes Lugar/Transição com capacidade infinita nos lugares. A rede Ghenesys é derivada da rede Condição/Evento, onde todos os arcos têm peso unitário e os elementos estáticos (condições) possuem capacidade de carregar apenas uma marca (ou, no caso de um box_capacidade, um número finito de marcas). Isso implica numa mudança na regra de habilitação das atividades: uma atividade está habilitada se seus boxes de entrada tiverem pelo menos uma marca e se o os boxes de saída ainda não atingiram seu limite máximo de marcas.

Para definir a regra de habilitação da rede Ghenesys em termos algébricos, vamos definir antes o produto direto de dois vetores:

Def: Dados dois vetores A e B de mesma dimensão m, o produto direto $C = [c_i] = A \otimes B$ é definido por $c_i = a_i \, . \, b_i$ para $1 \leq i \leq m$.

Seja a_k o vetor linha situado na linha k da matriz de incidência A. Este vetor, de dimensão $1 \times m$, descreve as pré-condições e as pós-condições da atividade e_k, isto é, $a_k = \bullet e_k + e_k \bullet$. O vetor $\bullet e_k$ corresponde a um vetor linha $1 \times m$, com o valor -1 na coluna que corresponde às pré-condições de e_k e o valor 0 nas demais colunas. Da mesma maneira, $e_k \bullet$ corresponde a um vetor linha $1 \times m$, com o valor 1 na coluna que corresponde às pós-condições de e_k e o valor 0 nas demais colunas.

Numa determinada marcação M, a condição dos boxes de entrada estarem marcados é dada por:

$$- M \otimes \bullet e_k \geq - \bullet e_k \qquad \text{(I)}$$

onde os sinais de negativo são para compensar o sinal negativo na definição de $\bullet e_k$ e a desigualdade reflete a possibilidade da marcação ser maior que um no caso do box_capacidade.

A outra condição, do número de marcas nos boxes de saída ser menor que sua respectiva capacidade, é dada por:

$$(C - M) \otimes e_k \bullet \geq e_k \bullet \qquad \text{(II)}$$

onde C é o vetor de capacidade dos boxes.

Estas alterações na habilitação das atividades não influem no cálculo de invariantes, pois a equação de estado e a matriz de incidência não são alteradas. Apenas altera-se o comportamento dinâmico da rede.

3.3.2 Box_montagem e Box_desmontagem

O atributo #relação_es da subclasse sub-rede-estática foi apresentado em 3.2.1. É utilizado por exemplo no caso de um objeto do tipo box_montagem (#relação_es = N:1) ou do tipo box_desmontagem (#relação_es = 1:N), para indicar a relação entre o número de marcas que entram e que saem do box. A existência deste atributo representa um problema para a análise de invariantes, porque marcas "surgem" ou "desaparecem" dentro dos boxes, sem que isto se reflita na equação de estado.

Uma solução para este problema seria substituir o box de montagem pela sub-rede-estática correspondente, como mostrado na figura abaixo, e calcular os invariantes na rede mais detalhada.

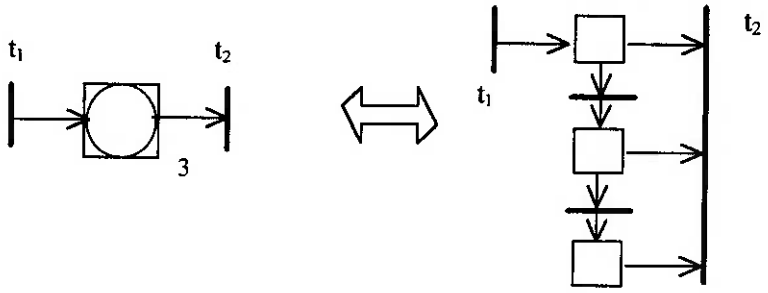


Figura 3.1 - Equivalência do box montagem com #relação_es 3:1

Outra solução é alterar a matriz de incidência e trabalhar como no caso das redes Lugar/Transição, com arcos de peso maior que um. No caso por exemplo de um box_montagem com #relação_es = N:1, o arco de entrada teria peso 1 (o segundo número da relação) e o arco de saída teria peso N (o primeiro número da relação).

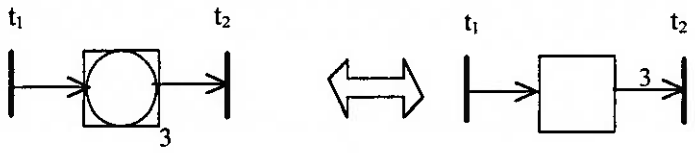


Figura 3.2 - Outra equivalência do box_montagem com #relação_es 3:1

Na análise num nível mais abstrato da rede, poderia ser utilizada a segunda alternativa, deixando a primeira alternativa para quando a análise estiver sendo feita num nível mais detalhado.

3.3.3 Portas (*Gates*) e Pseudo-condições

As portas [Peterson, 81] representam um tipo especial de relação de fluxo onde o disparo de uma transição não retira marcas do local de origem do sinal. Podem ser classificadas como porta habilitadora ou inibidora, dependendo do papel exercido pela marcação persistente de habilitar ou inibir uma transição.

Na modelagem de sistemas usando Ghenesys, sinais de controle podem se originar de elementos externos. Neste caso, estes elementos externos são modelados como pseudo-condições e, como estes elementos não pertencem ao sistema, são representados graficamente com linhas tracejadas ou preenchidas em cinza. Pela regra de habilitação de transição apresentado em 3.3.1, uma pseudo-condição pode ser interpretada como uma pré-condição da transição (no caso de uma porta habilitadora externa) ou como uma pós-condição da transição (no caso de uma porta inibidora externa). Exemplos de elementos externos ao sistema são: botões de reset, chaves liga/desliga, interrupções provocadas por sistemas de controle de nível superior e outros tipos de atuadores de controle.

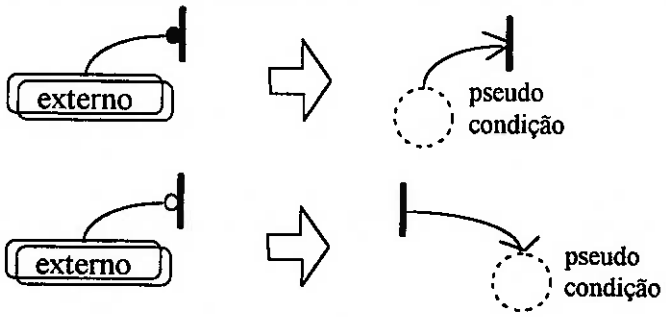


Figura 3.3 - Representação das pseudo-condições

A matriz de incidência de uma rede GHENESYS é, então, estendida para incluir todas as pseudo-condições e todas as portas externas. Chamando de A_n a matriz de incidência sem as pseudo-condições e de Π a matriz com as relações de fluxo com as pseudo-condições, a matriz de incidência A do sistema é dada por $[\ A_n \mid \Pi\ ]$.

Como as pseudo condições têm uma marcação persistente, é necessário devolver a marca depois do disparo da transição. Se representássemos esta devolução da marca como um arco, a rede deixaria de ser pura e não poderíamos mais utilizar a definição da matriz de incidência. Então, um termo Δ (matriz de reposição das marcas) é acrescentado à equação de estado para restaurar a marca. Esta matriz Δ, de mesma ordem da matriz de incidência A, é dada por $[\ 0 \mid \Pi\ ]$. Portanto, temos em A o comportamento do sistema incluindo as condições externas e em $\Delta = [\ 0 \mid \Pi\ ]$ a reposição das marcas nas pseudo-condições externas.

Assim, a equação de estado da rede Ghenesys fica:

$$M_{k+1} = M_k + (A^T - \Delta^T)\ \sigma_k.$$

Mantendo o termo Δ separado da matriz de incidência, a matriz de incidência A contém os termos necessários para achar as transições habilitadas na rede e as propriedades comportamentais.

A figura 3.4 abaixo mostra um exemplo (adaptado de [Silva J.R., 96]) de uma rede contendo portas e pseudo-condições.

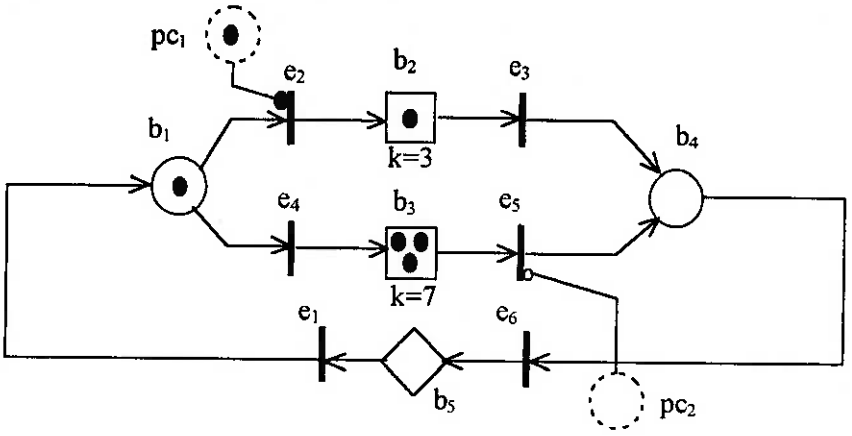


Figura 3.4 - Exemplo de rede

Sua matriz de incidência e matriz de reposição de marcas são

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 \end{bmatrix}$$

$$\Delta = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

A semelhança entre a equação de estado da rede Ghenesys e a equação das redes elementares (redes Lugar/Transição e Condição/Evento) faz com que os métodos de análise baseados na equação de estado sejam preservados. Por exemplo, a análise de invariantes na rede Ghenesys pode ser feita resolvendo a equação homogênea C.x = 0, onde, para S-invariantes a matriz C corresponde à diferença (A - Δ) na equação de estado, incluindo os gates e as pseudo-condições. O S-invariante x pode ser separado em componentes x_n , correspondente aos boxes normais, e termos x_p , correspondente às pseudo-condições. Então temos:

$$(A - \Delta) \,.\, \mathbf{x} = \mathbf{0} \quad \Leftrightarrow$$

$$([\, A_n \,|\, \Pi \,] - [\, \mathbf{0} \,|\, \Pi \,]) \,.\, [x_n \,|\, x_p \,]^T = \mathbf{0} \quad \Leftrightarrow$$

$$([\, A_n \,||\, \mathbf{0} \,] + [\, \mathbf{0} \,|\, \Pi \,] - [\, \mathbf{0} \,|\, \Pi \,]) \,.\, [x_n \,|\, x_p \,]^T = \mathbf{0} \quad \Leftrightarrow$$

$$[\, A_n \,|\, \mathbf{0} \,] \,.\, [x_n \,|\, x_p \,]^T = \mathbf{0} \quad \Leftrightarrow$$

$$\begin{cases} A_n \,.\, x_n = \mathbf{0} \\ \mathbf{0} \,.\, x_p = \mathbf{0} \end{cases}$$

Como já era esperado, os valores de x_p são indeterminados (a marcação das pseudo-condições não depende do sistema considerado) e portanto para se determinar os S-invariantes basta calcular x_n tal que $A_n \,.\, x_n = \mathbf{0}$

Por exemplo, na rede da figura 3.4, calculando em x_n o S-invariante mínimo do sistema é dado por x_n^T = (1 1 1 1 1). Acrescentando os termos x_p, os S-invariantes mínimos são: x_1^T = (1 1 1 1 1 0 0), x_2^T = (0 0 0 0 0 1 0), x_3^T = (0 0 0 0 0 0 1) e qualquer combinação linear destes vetores é um S-invariante.

Para o cálculo dos T-invariantes da rede, usa-se a matriz de incidência transposta. Então temos:

$$(A - \Delta)^T \cdot \mathbf{y} = \mathbf{0} \quad \Leftrightarrow$$

$$([\, A_n \,|\, \Pi\,] - [\, \mathbf{0} \,|\, \Pi\,])^T \cdot \mathbf{y} = \mathbf{0} \quad \Leftrightarrow$$

$$([\, A_n \,|\, \mathbf{0}\,] + [\, \mathbf{0} \,|\, \Pi\,] - [\, \mathbf{0} \,|\, \Pi\,])^T \cdot \mathbf{y} = \mathbf{0} \quad \Leftrightarrow$$

$$[\, A_n \,|\, \mathbf{0}\,]^T \cdot \mathbf{y} = \mathbf{0} \quad \Leftrightarrow$$

$$\begin{cases} A_n{}^T \cdot \mathbf{y} = \mathbf{0} \\ \mathbf{0} \cdot \mathbf{y} = \mathbf{0} \end{cases}$$

Ou seja, estamos apenas adicionando equações que são sempre verdadeiras e não alteram as soluções do sistema. Basta calcular os T-invariantes usando a matriz $A_n{}^T$. Voltando ao exemplo da figura 3.4, os T-invariantes mínimos do sistema são $y_1{}^T = (1\ 1\ 1\ 0\ 0\ 1)$ e $y_2{}^T = (1\ 0\ 0\ 1\ 1\ 1)$.

3.3.4 Atividade Composta

Uma importante característica da rede Ghenesys é a possibilidade de se fazer a síntese da rede misturando as abordagem *top-down* e *bottom-up* (característico de sistemas de manufatura), através de refinamentos sucessivos e modularidade. Diferentemente da proposta original do PFS/MFG (que é um método para se elaborar uma rede de Petri MFG), o Ghenesys pode ser simulado em qualquer nível de abstração. Isso é possível através de objetos da subclasse atividade_composta (ou sub-rede dinâmica).

Por exemplo, a figura abaixo mostra um refinamento da atividade b_5 da figura 3.4

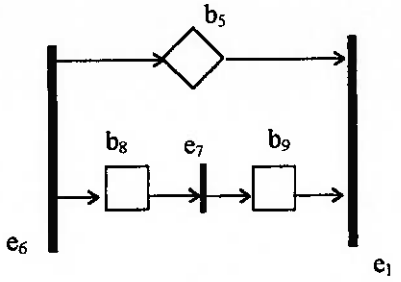


Figura 3.5 - Exemplo de refinamento

Utilizando esta forma estruturada de refinamento, é possível dividir o problema de cálculo de invariantes em matrizes menores. Vejamos o que acontece com a matriz de incidência quando se faz o refinamento de uma atividade_composta. Sem perda de generalidade, podemos supor que o box_temporizado e as atividades de entrada e saída sejam os últimos elementos na matriz de incidência (se não forem, pode-se sempre criar um grafo equivalente trocando-se apenas os nomes dos vértices). Podemos, assim, particionar a matriz de incidência (antes do refinamento) da seguinte forma, com as respectivas dimensões:

$$A = \begin{array}{c} \begin{matrix} m-1 & 1 \end{matrix} \\ \left[\begin{array}{cc} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right] \begin{array}{l} n-2 \\ \hline 2 \end{array} \end{array}$$

A última coluna da matriz de incidência representa as relações de fluxo entre o box_temporizado e as atividades da rede. No caso, o box_temporizado tem apenas uma atividade de entrada e uma de saída, então $A_{12} = \mathbf{0}$ e $A_{22} = [\ 1\ \ -1\ ]^T$.

Após o refinamento, a matriz de incidência torna-se:

$$A' = \begin{array}{c} \begin{matrix} m-1 & 1 & m' \end{matrix} \\ \left[\begin{array}{ccc} A_{11} & A_{12} & A_{13} \\ \hline A_{21} & A_{22} & A_{23} \\ \hline A_{31} & A_{32} & A_{33} \end{array}\right] \begin{array}{l} n-2 \\ \hline 2 \\ \hline n' \end{array} \end{array}$$

onde m' e n' representam o número de boxes e atividades na subrede refinada, respectivamente.

Os novos elementos que surgem com o refinamento apenas se relacionam entre si ou com a atividade de entrada e saída da atividade_composta. Então $A_{13} = \mathbf{0}$, $A_{31} = \mathbf{0}$ e $A_{32} = \mathbf{0}$. Logo, a matriz de incidência torna-se:

$$A' = \begin{bmatrix} A_{11} & \mathbf{0} & \mathbf{0} \\ A_{21} & A_{22} & A_{23} \\ \mathbf{0} & \mathbf{0} & A_{33} \end{bmatrix}$$

S-invariantes

Tendo a matriz de incidência, vejamos o que acontece com os S-invariantes após o refinamento. Seja $\mathbf{x} = [\ \mathbf{x}_1 \ ¦\ k\ ]$ um S-invariante da rede Ghenesys antes do refinamento, onde k é um número inteiro associado ao box_temporizado. Então

$$A \,.\, \mathbf{x} = 0 \quad \Leftrightarrow$$

$$\begin{cases} A_{11} \,.\, \mathbf{x}_1 = 0 \\ A_{21} \,.\, \mathbf{x}_1 + A_{22} \,.\, k = 0 \end{cases}$$

Na rede refinada, a extensão deste S-invariante com zeros $\mathbf{x}' = [\ \mathbf{x}_1 \ ¦\ k \ ¦\ \mathbf{0}\ ]$ ainda é um S-invariante. De fato:

$$A' \cdot \mathbf{x}' = \begin{bmatrix} A_{11} \cdot \mathbf{x}_1 \\ A_{21} \cdot \mathbf{x}_1 + A_{22} \cdot k + A_{23} \cdot \mathbf{0} \\ A_{33} \cdot \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

Ou seja: neste desenvolvimento estruturado, os S-invariantes da rede em nível mais alto continuam sendo invariantes da rede após uma extensão.

Além destes invariantes que são preservados, novos S-invariantes podem surgir. Assim, é necessário calcular os S-invariantes dentro da atividade_composta. Seja $\mathbf{x}''= [\, k \,|\, \mathbf{x}_2 \,]$ uma solução da seguinte equação matricial:

$$\begin{bmatrix} -A_{22} & A_{23} \\ \mathbf{0} & A_{33} \end{bmatrix} \cdot \mathbf{x}'' = 0 \Leftrightarrow \begin{vmatrix} -A_{22} \cdot k + A_{23} \cdot \mathbf{x}_2 = 0 \\ A_{33} \cdot \mathbf{x}_2 = 0 \end{vmatrix}$$

A coluna $[\, A_{23} \,|\, A_{33} \,]^T$ corresponde à matriz de incidência da subrede obtida da expansão da atividade. Como esta subrede não é cíclica, se aplicássemos o algoritmo para cálculo de invariantes nesta matriz poderíamos não achar S-invariante algum. Por isso, para o cálculo de S-invariantes, foi acrescentado um box auxiliar, complementar do box_temporizado, que representa a influência do resto da rede sobre esta subrede, representado pela matriz $-A_{22}$. Para encontrar $\mathbf{x}''$, podemos novamente usar os algoritmos do capítulo 2, aplicados à matriz acima.

Compondo $\mathbf{x}'$ e $\mathbf{x}''$ obtemos os novos S-invariantes da rede expandida. De fato, o vetor $\mathbf{x} = [\, \mathbf{x}_1 \,|\, 0 \,|\, \mathbf{x}_2 \,]$ também é um S-invariante, pois:

$$A \cdot \mathbf{x} = \begin{bmatrix} A_{11} \cdot \mathbf{x}_1 \\ A_{21} \cdot \mathbf{x}_1 + A_{23} \cdot \mathbf{x}_2 \\ A_{33} \cdot \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} A_{11} \cdot \mathbf{x}_1 \\ A_{22} \cdot k - A_{22} \cdot k \\ A_{33} \cdot \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

Portanto, os passos para o cálculo dos S-invariantes são:

- Calcular os S-invariantes na rede de nível superior. Como foi mostrado acima, um S-invariante calculado num nível superior também é um S-invariante após o detalhamento.

- Calcular os S-invariantes dentro da subrede, invertendo o sinal da matriz A_{22}, isto é, trocando o box_temporizado pelo seu complementar. Esta inversão de sinal pode ser explicada pelo fato do detalhamento de uma atividade gerar uma sub-rede não-cíclica. Substituindo pelo complementar, voltamos a ter uma rede cíclica, podendo utilizar os algoritmos apresentados no capítulo anterior.
- Compor (somar) os S-invariantes encontrados.

Por exemplo, no caso da expansão da atividade no exemplo acima, temos:

Passo 1: S-Invariantes da rede de nível superior. Como já visto no item anterior, e deixando de lado os invariantes correspondentes às pseudo-condições, o único invariante mínimo é x_1 = (1 1 1 1 1 0 0).

Passo 2: S-Invariante na subrede: x_2 = (1 1 1) onde os fatores correspondem aos boxes b_5, b_8 e b_9, respectivamente.

Passo 3: Composição e extensão dos S-invariantes: na rede completa, os S-Invariantes mínimos são:

Extensão de x_1, completando com zeros: (1 1 1 1 1 0 0 0 0 0)

Composição x_1 , x_2 (zerando b_5 e acrescentando x_2): (1 1 1 1 0 0 0 1 1)

T-invariantes

Um método similar pode ser utilizado para o cálculo dos T-invariantes da rede, agora trabalhando sobre a matriz de incidência transposta:

$$A'^{T}=\begin{bmatrix} A_{11}^{T} & A_{21}^{T} & \mathbf{0} \\ \mathbf{0} & A_{22}^{T} & \mathbf{0} \\ \mathbf{0} & A_{23}^{T} & A_{33}^{T} \end{bmatrix}$$

Seja $\mathbf{y} = [\ \mathbf{y}_1 \ ¦\ \mathbf{k}\ ]$ um T-invariante da rede Ghenesys antes do refinamento, com $\mathbf{k} = [\ k_1\ k_2\ ]$ correspondendo aos eventos de entrada e de saída da atividade composta. Então

$$A^T \,.\, \mathbf{y} = 0 \quad \Leftrightarrow$$

$$\begin{cases} A_{11}{}^T \,.\, \mathbf{y}_1 + A_{21}{}^T \,.\, \mathbf{k} = 0 \\ A_{22}{}^T \,.\, \mathbf{k} = 0 \end{cases}$$

Como $A_{22}{}^T = [\ -1\ 1\ ]$, então da segunda condição temos

$$-k_1 + k_2 = 0 \quad \Leftrightarrow \quad k_1 = k_2$$

Este resultado mostra que, em qualquer T-invariante, o fator correspondente aos eventos de entrada e de saída devem ser os mesmos. De fato, eles são os únicos eventos a colocar e retirar marcas do box_temporizado, então um precisa compensar o outro. Podemos substituir k_1 e k_2 por uma constante arbitrária w ($\mathbf{k} = [\ w\ w\ ]$).

Após o refinamento da atividade, a extensão com zeros de um T-invariante $\mathbf{y} = [\ \mathbf{y}_1\ ¦\ \mathbf{k}\ ¦\ 0\ ]$, com $\mathbf{k} \neq 0$, não é mais um T-invariante, pois:

$$A'^{T}\cdot\mathbf{y} = \begin{bmatrix} A_{11}{}^{T}\cdot\mathbf{y}_1 + A_{21}{}^{T}\cdot\begin{bmatrix} w & w \end{bmatrix}^{T} \\ A_{22}{}^{T}\cdot\begin{bmatrix} w & w \end{bmatrix}^{T} \\ A_{23}{}^{T}\cdot\begin{bmatrix} w & w \end{bmatrix}^{T} + A_{33}{}^{T}\cdot\mathbf{0} \end{bmatrix}$$

e a última linha pode ser diferente de zero. A interpretação deste resultado é que, após o refinamento, o evento de entrada da atividade_composta coloca marcas em alguns dos boxes da subrede expandida e o evento de saída não retira as marcas destes mesmos boxes. Se $\mathbf{k} \neq 0$, é preciso considerar também as atividades de dentro da subrede.

Para calcular os T-invariantes dentro da subrede, desta vez não é necessário substituir o box_temporizado pelo seu complementar. Seja $\mathbf{y}$" um dos T-invariantes encontrados, dado por $\mathbf{y}$" = [$\mathbf{k}$ ¦ $\mathbf{y}_2$]. Então temos:

$$\begin{bmatrix} A_{22}^T & 0 \\ A_{23}^T & A_{33}^T \end{bmatrix} \cdot \mathbf{y} = 0 \Leftrightarrow \begin{vmatrix} A_{22}^T \cdot \begin{bmatrix} k & k \end{bmatrix}^T = 0 \\ A_{23}^T \cdot \begin{bmatrix} k & k \end{bmatrix}^T + A_{33}^T \cdot \mathbf{y_2} = 0 \end{vmatrix}$$

Os T-invariantes da rede completa são dados pelos vetores $\mathbf{y}$ = [$\mathbf{y}_1$ ¦ k ¦ $\mathbf{y}_2$], pois:

$$A'^T \cdot \mathbf{y} = \begin{bmatrix} A_{11}^T \cdot \mathbf{y_1} + A_{21}^T \cdot \begin{bmatrix} k & k \end{bmatrix}^T \\ A_{22}^T \cdot \begin{bmatrix} k & k \end{bmatrix}^T \\ A_{23}^T \cdot \begin{bmatrix} k & k \end{bmatrix}^T + A_{33}^T \cdot \mathbf{y_2} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

Portanto, os passos para o cálculo dos T-invariantes são:

- Calcular os T-invariantes na rede de nível superior.
- Calcular os T-invariantes dentro da subrede.
- Os T-invariantes da rede são obtidos da composição dos T-invariantes encontrados.

Voltando novamente ao exemplo do refinamento da atividade das figuras 3.4 e 3.5, temos:

Passo 1: T-Invariantes da rede de nível superior. Como já visto no item anterior, os T-invariante mínimo são $y_1 = (1\ 1\ 1\ 0\ 0\ 1)$ e $y_2 = (1\ 0\ 0\ 1\ 1\ 1)$.

Passo 2: T-Invariante na subrede: $y_3 = (1\ 1\ 1)$, onde os fatores correspondem às atividades e_1, e_6 e e_7, respectivamente.

Passo 3: Composição dos T-invariantes: na rede completa, os T-Invariantes mínimos são:

Composição y_1 , y_3 : (1 1 1 0 0 1 1)

Composição y_2 , y_3 : (1 0 0 1 1 1 1)

3.4 Conclusão

Os exemplos mostrados neste capítulo foram simples, (expansão de uma única atividade, um nível hierárquico, etc.), mas, respeitada a estruturação, o procedimento para cálculo de invariantes pode ser generalizado para mais de uma atividade ou mais de um nível hierárquico, como será mostrado no próximo capítulo.

Para o caso das pseudo-condições, foi mostrado que os cálculos podem ser feitos sem considerá-los e, no caso de S-invariantes, apenas acrescentar algumas outras soluções (triviais) às soluções já encontradas.

O método de invariantes pode assim ser aplicado para a rede Ghenesys, utilizando-se dos mesmos algoritmos para cálculo apresentados no capítulo anterior.

Com uma importante vantagem: a possibilidade de se dividir hierarquicamente o problema de se achar os semiflows, o que é uma vantagem visto que este é um problema de complexidade exponencial.

CAPÍTULO 4 - RESULTADOS

4.1 Generalização do Método

Os cálculos apresentados no capítulo anterior foram feitos a partir de exemplos simples. Pode-se fazer, no entanto, uma generalização do método para cálculo de invariantes na rede Ghenesys, aproveitando a forma estruturada da rede:

Calcular os invariantes na rede de nível superior

Repetir
Se necessário, incluir as pseudo-condições e substituir boxes montagem (ou desmontagem) pela subrede equivalente
Calcular invariantes na subrede expandida
Compor os invariantes encontrados no nível superior e na subrede
Até o nível desejado

onde o cálculo de invariantes deve ser feito usando um dos algoritmos apresentados no capítulo 2 e a composição dos invariantes deve ser feita segundo a seguinte regra:

S-invariantes

- para os invariantes encontrados no nível superior, completar com zeros as posições correspondentes aos boxes na subrede;

- para os invariantes encontrados na subrede detalhada e cujo fator correspondente ao box_temporizado for igual a zero, completar com zeros nas posições correspondentes aos boxes do nível superior;

- para os invariantes encontrados cujo fator correspondente ao box_temporizado for diferente de zero, zerar a posição correspondente ao

box_temporizado e, nos demais boxes, usar o fator encontrado no invariante da subrede correspondente.

T-invariantes

- para os invariantes encontrados no nível superior, cujos fatores correspondentes às atividades de entrada e saída for igual a zero, completar com zeros nas posições correspondentes às atividades na subrede;

- para os invariantes encontrados na subrede detalhada, cujos fatores correspondente às atividades de entrada e saída for igual a zero, completar com zeros nas posições correspondentes às atividades do nível superior;

- para os invariantes encontrados cujos fatores correspondentes às atividades de entrada e saída for diferente de zero, usar o fator encontrado no invariante da subrede correspondente.

A grande vantagem do cálculo de invariantes na rede Ghenesys é a possibilidade de se fazer o cálculo hierarquicamente. Com isso, é possível dividir a matriz de incidência em submatrizes de menor tamanho, o que é uma grande vantagem visto que o problema de se achar os invariantes de componentes inteiros não-negativos é de complexidade exponencial. Além disso, se a análise num nível hierárquico superior já for conclusiva, não é necessário realizar os cálculos até o último nível de detalhe.

No item seguinte, serão apresentados os argumentos que justificam a possibilidade desta generalização. Por fim, será apresentado o cálculo de invariantes num

exemplo de modelagem de sistema de manufatura em rede Ghenesys, adaptado de [Miyagi88].

4.2 Justificativa

4.2.1 Posição das linhas e colunas na matriz de incidência

Nos cálculos apresentados no capítulo anterior, as últimas linhas da matriz de incidência estavam relacionadas às atividades de entrada e saída da atividade_composta e as últimas colunas estavam relacionadas ao box de tempo e às pseudo-condições. Mesmo quando isso não ocorre, os resultados continuam válidos.

A justificativa é porque sempre é possível obter um grafo equivalente [Reisig, 85], apenas trocando os nomes dos elementos do grafo, de tal forma que as condições acima sejam satisfeitas. Os cálculos então seriam feitos nesta rede equivalente. Da mesma forma, ao final é possível rearranjar os invariantes encontrados, de tal forma que os componentes se refiram aos boxes ou atividades na ordem inicial.

4.2.2 Número de Atividades

O exemplo apresentado foi para a expansão de uma única atividade. O resultado, no entanto, continua válido mesmo para um número maior de atividades ou mesmo para o caso de mais de um nível de detalhamento, desde que seja feito o detalhamento de uma atividade por vez. A demonstração é por indução.

Passo 1:

Para a expansão de uma atividade, já foi mostrado que o resultado é válido.

Passo N:

Supondo que o resultado seja válido após a expansão de N-1 atividades, mostrar que continua válido na expansão seguinte.

Chamando a matriz de incidência após o passo N-1 de A_{N-1} e utilizando o resultado do item 4.2.1, é possível rearranjar A_{N-1} de forma que as últimas linhas da matriz de incidência estejam relacionadas às atividades de entrada e saída da atividade a ser expandida, e a última coluna esteja relacionada ao box_temporizado. Após a expansão, os argumentos utilizados para determinar que algumas das submatrizes da nova matriz de incidência sejam identicamente nulas continuam verdadeiros. Com isso, o resto da demonstração é igual ao do passo 1.

4.3 Software

Para o cálculo de invariantes na rede Ghenesys, foram desenvolvidas 3 funções utilizando a linguagem de programação do software MAPLE V. Abaixo está uma breve descrição destas funções, a listagem completa destas funções está no apêndice D.

1. Cálculo de Invariantes

Esta função implementa o algoritmo para cálculo de invariantes mínimos apresentado no capítulo 2. O parâmetro de entrada é uma matriz, e a saída é uma matriz cujas linhas contêm os invariantes da rede. Se não existir nenhum invariante não-nulo na rede, a função devolve uma linha de zeros.

Foi implementado o algoritmo apresentado em 2.2.2, porém apenas com parte das heurísticas apresentadas. A seguir está a parte principal desta função:

```
#para cada coluna a ser anulada
for i to n do
# separação das linhas em 2 matrizes Up e Un
for j from 2 to m do
  if U0[j,i] = 0 then     U1 := stack(U1, row(U0,j))
```

```
    elif U0[j,i] <0 then  Un := stack(Un, row(U0,j))
    else Up := stack(Up, row(U0,j))
  fi
 od:
# União das linhas em Up e Un
  for k from 2 to rowdim(Un) do
    for kk from 2 to rowdim(Up) do
      X := abs(Up[kk,i])*row(Un, k) + abs(Un[k,i])*row(Up, kk):
      U1 := stack(U1, X)
    od
  od
od:
# retirando as colunas não-nulas de U1, que correspondem aos Invariantes
if rowdim(U1) > 1 then
  U1:=submatrix(U1, 2..rowdim(U1),n+1..coldim(U1)):
else
  U1:=submatrix(Linha_Zeros, 1..1,n+1..coldim(Linha_Zeros))
fi:
```

2. S_Compor

Esta função faz a composição dos S-invariantes conforme a regra estabelecida no item 4.1. Possui como parâmetros de entrada: a matriz com os invariantes do nível superior, o índice do box_temporizado que está sendo expandido e a matriz com os invariantes da subrede detalhada (a primeira coluna desta matriz deve corresponder ao box_temporizado da atividade que está sendo expandida). A saída é uma matriz cujas linhas contêm os invariantes da rede. A seguir está a parte principal desta função:

```
# extensão dos invariantes de nível superior
U1 := extend(X1,0,coldim(X2)-1,0):
D := array(sparse, 1..1,1..coldim(X1)):
# acha invariantes cujos coeficientes do box temporizado são > 0
for i from 1 to rowdim(X1) do
  if X1[i,Pos1] > 0
    then D := stack(D, row(X1,i))
  fi
od:
m := rowdim(D):

for i from 1 to rowdim(X2) do
  if X2[i,1] = 0
```

```
    then X:=augment(array(sparse, 1..1,1..coldim(X1)), submatrix(X2, i..i,
2..coldim(X2))):
      U1:=stack(U1, X):
    else for j from 2 to m do
      # composição dos invariantes
      AUX1:=submatrix(D, j..j, 1..coldim(D)):
      AUX2:=submatrix(X2, i..i, 2..coldim(X2)):
      AUX1:=mulrow(AUX1,1,X2[i,1]):
      AUX2:=mulrow(AUX2,1,D[j,Pos1]):
      X:=augment(AUX1, AUX2):
      X[1,Pos1]:=0:
      U1:=stack(U1, X):
     od:
  fi
od:
```

3. T_Compor

Esta função faz a composição dos T-invariantes conforme a regra estabelecida no item 4.1. Possui como parâmetros de entrada: a matriz com os invariantes do nível superior, o índice correspondente à atividades de entrada e a matriz com os invariantes da subrede detalhada (as duas primeiras colunas desta matriz devem corresponder às atividades de entrada e saída da atividade composta, respectivamente). A saída é uma matriz cujas linhas contêm os invariantes da rede. A seguir está a parte principal desta função:

```
D := array(sparse, 1..1,1..coldim(Y1)):
U1 := array(sparse, 1..1,1..coldim(Y1)+coldim(Y2)-2):
for i from 1 to rowdim(Y1) do
  if Y1[i,Pos1] > 0
     then D := stack(D, row(Y1,i)):
     else X:=augment(submatrix(Y1, i..i, 2..coldim(Y1)), array(sparse,
1..1,1..coldim(Y2)-2)):
      U1:=stack(U1, X):
  fi
od:
m := rowdim(D):

for i from 1 to rowdim(Y2) do
  if Y2[i,1] = 0
```

```
    then X:=augment(array(sparse, 1..1,1..coldim(Y1)), submatrix(Y2, i..i,
3..coldim(Y2))):
      U1:=stack(U1, X):
    else for j from 2 to m do
      X:=augment(Y2[i,1]*submatrix(D, j..j, 1..coldim(D)), D[j,Pos1]*submatrix(Y2,
i..i, 3..coldim(Y2))):
      U1:=stack(U1, X):
     od:
  fi
od:
if rowdim(U1) > 1 then
  RETURN(submatrix(U1, 2..rowdim(U1),1..coldim(U1)))
 else
  RETURN(submatrix(U1, 1..1,1..coldim(U1)))
fi:
```

O programa principal com um exemplo de utilização destas rotinas é dado a seguir:

```
### leitura do arquivo com a definição das funções e entrada da matriz
read `/maplev3/invar.txt`:
read `/maplev3/entrada.txt`:
XX := Invariante(transpose(A)):
YY := Invariante(A):

### repetir para cada atividade a ser expandida
read `/maplev3/entrada2.txt`:
# gerar coluna extra com o complementar do box_temporizado
# apenas para o cálculo de S-Invariantes
Coluna:=array(sparse,1..rowdim(A1),1..1):
Coluna[1,1]:=-1: Coluna[2,1]:=1:
C:=augment(Coluna,A1):
XX1:=Invariante(transpose(C)):
YY1:=Invariante(A1):
### Alterar próxima linha conforme o índice do box_temporizado
### e da atividade de entrada
Box_Tempo:=3: Ativ_Entrada:=3:
XX:=S_Compor(XX,Box_Tempo,XX1):
YY:=T_Compor(YY,Ativ_Entrada, YY1):
```

4.4 Exemplo

4.4.1 Descrição do sistema

O exemplo que será utilizado foi adaptado de [Miyagi, 88]. Corresponde a um sistema de manufatura, composto por uma unidade de montagem, uma máquina para processamento de material, uma unidade de desmontagem e um robô para carregamento e descarregamento.

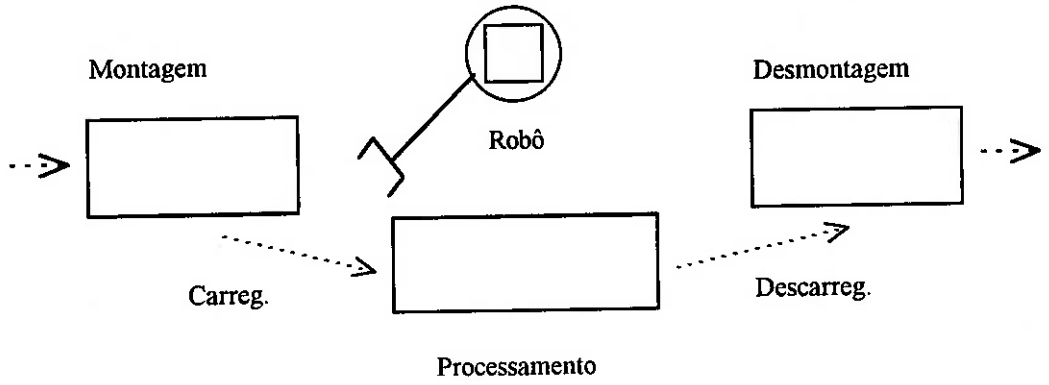


Figura 4.1 - Lay-out do sistema

O modelo do sistema em rede Ghenesys está na figura 4.2. Por simplificação, o box_montagem foi considerado como tendo #relação_es 5:1 (montagem de um conjunto de 5 peças) e o box_desmontagem como 1:5.

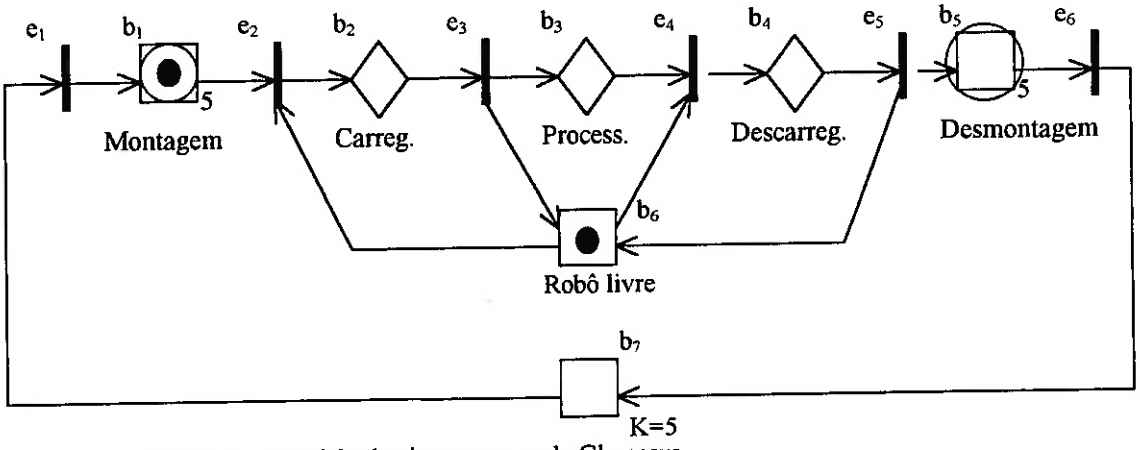


Figura 4.2 - Modelo do sistema em rede Ghenesys

Utilizando a equivalência do box de montagem e de desmontagem que altera o peso do arco, a matriz de incidência é dada por:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ -5 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 5 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}$$

Para validar o método de cálculo de invariantes mostrado em 4.1, duas atividades da rede da figura 4.2 serão expandidas em subredes, sendo que uma das subredes contém ainda uma atividade que será também expandida (para mostrar que pode-se trabalhar com mais de um nível hierárquico).

A operação de Carregamento poderia ser expandida nas fases de Agarrar, Transportar e Soltar, conforme a figura abaixo:

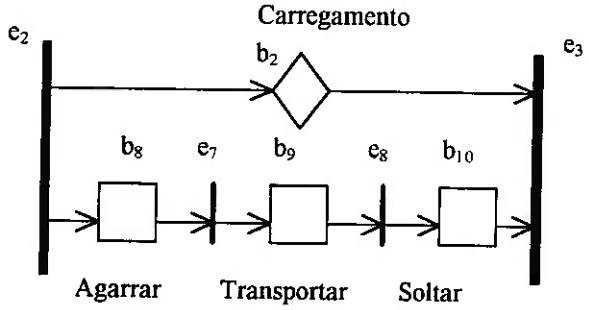


Figura 4.3 - Subrede 1: Expansão da atividade Carregamento

Sem considerar o box_temporizado, a matriz de incidência da subrede da figura 4.3 é dada por:

$$A' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix}$$

A figura 4.4 apresenta a expansão da atividade Processamento, supondo no exemplo que há duas possibilidades de processos, e a figura 4.5 apresenta a expansão da atividade correspondente ao box b_{13} da figura 4.4.

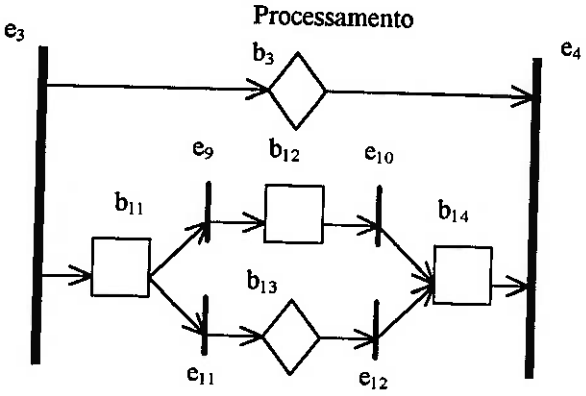


Figura 4.4 - Subrede 2: Expansão da atividade Processamento

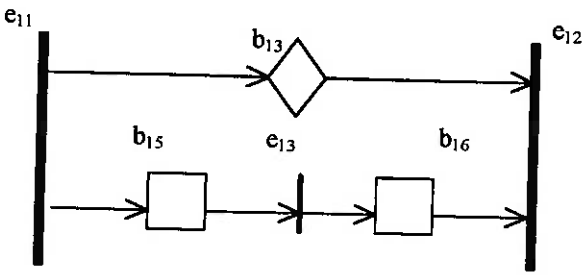


Figura 4.5 - Subrede 3: Expansão da atividade b_{13}

As matrizes de incidência das subredes 2 e 3 são dadas, respectivamente, por:

$$A' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \qquad A' = \begin{bmatrix} 1 & 0 \\ 0 & -1 \\ -1 & 1 \end{bmatrix}$$

4.4.2 Invariantes do Sistema

Usando o programa desenvolvido em MAPLE, foram encontrados os T- e S-invariantes do sistema em cada subrede.

Nível superior:

- S-Invariantes:

(0 1 0 1 0 1 0) - indica que o robô está em um dos estados: Livre, Carregando ou Descarregando.

(1 5 5 5 1 0 1) - mostra o fluxo de material e a conservação de marcas durante o processamento.

Somando estes dois invariantes, obtemos um invariante positivo. Portanto, o sistema é estruturalmente limitado e conservativo.

- T-Invariante:

(5 1 1 1 1 5) - mostra que após o disparo das atividades 1 e 6 (5 vezes cada) e das atividades 2 a 5 (1 vez cada), o sistema volta para o estado inicial. Ou seja, o sistema é cíclico e repetitivo.

Subrede 1:

- S-Invariante (1° termo corresponde ao complementar do box_temporizado):

(1 1 1 1) - mostra que dentro da atividade composta, o sistema está apenas em um dos estados internos (Agarrar, Transportar ou Soltar).

- T-Invariante (dois termos iniciais correspondem às atividades de entrada e saída):

(1 1 1 1) - mostra que após um disparo de cada atividade, o sistema volta para o estado inicial.

Da mesma formam, na subrede 2 o único S-Invariante é (1 1 1 1 1) e os T-Invariantes são (1 1 1 1 0 0) e (1 1 0 0 1 1). Na subrede 3, o S-Invariante é (1 1 1) e o T-Invariante é (1 1 1).

Compondo estes invariantes para a rede completa segundo a regra apresentada anteriormente, obtemos:

S-Invariantes:

(0 1 0 1 0 1 0 0 0 0 0 0 0 0 0)
(1 5 5 5 1 0 1 0 0 0 0 0 0 0 0)
(0 0 0 1 0 1 0 1 1 1 0 0 0 0 0)
(1 0 5 5 1 0 1 5 5 5 0 0 0 0 0)
(1 5 0 5 1 0 1 0 0 0 5 5 5 5 0 0)
(1 0 0 5 1 0 1 5 5 5 5 5 5 5 0 0)
(1 5 0 5 1 0 1 0 0 0 5 5 0 5 5 5)
(1 0 0 5 1 0 1 5 5 5 5 5 5 0 5 5 5)

Observe que os dois primeiros S-Invariantes são os mesmos invariantes do nível superior, completados com zeros, portanto possuem a mesma interpretação já apresentada para a rede de nível superior.

T-invariantes:

(5 1 1 1 1 5 1 1 1 1 0 0 0)

(5 1 1 1 1 5 1 1 0 0 1 1 1)

Estes invariantes mostram o número de disparos necessários de cada atividade da rede para que o sistema volte ao estado inicial.

Para verificar a correção das operações efetuadas, foram calculados também os invariantes usando a matriz de incidência da rede completa (uma matrix de tamanho 13 × 16) com o mesmo programa em MAPLE. O tempo gasto para encontrar estes invariantes foi de 115 segundos num microcomputador 486DX2/66. Dividindo o problema em subredes, o tempo total de processamento, incluindo a composição dos invariantes foi de 29 segundos.

Portanto, esta forma estruturada e hierárquica permite obter os invariantes da rede, facilita a interpretação dos invariantes em cada subrede e ainda reduz o tempo de processamento.

CAPÍTULO 5 - CONCLUSÃO

Foi apresentado que a análise de invariantes pode ser utilizada para verificação de propriedades da rede Ghenesys, apesar das diferenças existentes entre seu formalismo matemático e o da rede Lugar/Transição.

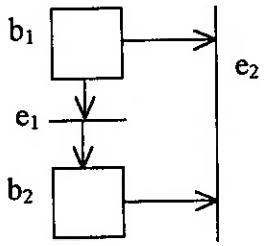
Foi mostrado também que, utilizando uma forma estruturada para o desenvolvimento da rede, também o cálculo de invariantes pode ser feito por partes, hierarquicamente, o que é uma vantagem em termos de método de desenvolvimento e também em termos computacionais. Não é possível fazer uma estimativa da ordem de grandeza da redução no tempo de processamento para o caso de uma rede qualquer.

Trabalhos futuros:

- A rede Ghenesys ainda está em desenvolvimento. Uma idéia é de se introduzir a superclasse *marca*, de tal forma que os atributos desta superclasse suportem a caracterização das marcas como na rede Colorida [Jensen, 90]. Com esta alteração, é preciso reestudar a definição do método de invariantes.

- Analisar a questão da unimodularidade da matriz de incidência. Uma matriz é totalmente unimodular (TUM) quando o determinante de qualquer uma de suas submatrizes quadradas for igual a 0, +1 ou -1 [Schrijiver, 86]. Uma condição necessária para uma matriz ser TUM é que cada elemento da matriz seja igual a 0, -1 ou +1, o que acontece com a matriz de incidência da rede Ghenesys. No entanto, se houver estruturas

como as da figura abaixo, o determinante da submatriz é igual a 2 e a matriz de incidência deixa de ser TUM. Resta verificar em quais casos a matriz é TUM e quais as vantagens (se houver alguma) para o cálculo de invariantes se a matriz for TUM.



$$A = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}$$

$\det(A) = 2$

Figura 5.1 - Exemplo de não unimodularidade

REFERÊNCIAS BIBLIOGRÁFICAS

[Barkaoui, 92] BARKAOUI, K; MINOUX, M. A Polynomial-Time Graph Algorithm to Decide Liveness of Some Basic Classes of Bounded Petri Nets, Application and Theory of Petri Nets 1992, Lecture Notes in Computer Science vol. 616, Springer-Verlag, Berlin, 1992.

[Bernardinello, 92] BERNARDINELLO, L. e DE CINDIO, F. A Survey of Basic Net Models and Modular Net Classes, Lecture Notes In Computer Science vol. 609, Springer-Verlag, Berlin, 1992.

[Cao, 90] CAO, X.R. e HO Y.C. Models of Discrete Event Dynamics Systems, IEEE Control Systems Magazine, vol 10, nº 4, IEEE, 1990.

[Cheb-Terrab, 93] CHEB-TERRAB, E.S. Curso de Computação Algébrica em MAPLE V, Notas de Aula, Universidade Estadual do Rio de Janeiro, Rio de Janeiro, 1993.

[Christensen, 92] CHRISTENSEN, S. e PETRUCCI, L. Towards a Modular Analysis of Coloured Petri Nets, Application and Theory of Petri Nets 1992, Lecture Notes In Computer Science vol. 616, Springer-Verlag, Berlin, 1992.

[Colom, 90a] COLOM, J.M. e SILVA, M. Convex Geometry and Semiflows in P/T nets. A Comparative Study of Algorithms for Computation of Minimal P-Semiflows, Advances in Petri Nets 1990, Lecture Notes In Computer Science vol. 483, Springer-Verlag, Berlin, 1990.

[Colom, 90b] COLOM, J.M. e SILVA, M. Improving the Linearly Based Characterization of P/T Nets, Advances in Petri Nets 1990, Lecture Notes In Computer Science vol. 483, Springer-Verlag, Berlin, 1990.

[Colom, 90c] COLOM, J.M.; CAMPOS, J. e SILVA, M. On Liveness Analysis through Linear Algebraic Techniques. In Proc. of the Annual General Meeting of ESPRIT Basic Research Action, Paris, 1990.

[Couvreur, 90] COUVREUR, J.M. e MARTÍNEZ, J. Linear Invariants in Commutative High Level Nets, Advances in Petri Nets 1990, Lecture Notes In Computer Science vol. 483, Springer-Verlag, Berlin, 1990.

[Couvreur, 93] COUVREUR J.M.; HADDAD, S. e PEYRE, J.F. Generative Families of Positive Invariants in Coloured Nets Sub-Classes, Advances in Petri Nets 1993, Lecture Notes In Computer Science vol. 674, Springer-Verlag, Berlin, 1993.

[Desel, 94] DESEL, J. e RADOLA, M.D. Proving Non-Reachability by Modulo-Place-Invariants, Foundations of Software Technology and Theoretical Computer Science , Lecture Notes In Computer Science vol. 880, Springer-Verlag, Berlin, 1994.

[Ezpeleta, 93] EZPELETA, J.; COUVREUR, J.M.; SILVA, M. A New Technique for Finding a Generating Family of Siphons, Traps and ST-Components. Application to Colored Petri Nets, Advances in Petri Nets 1993, Lecture Notes In Computer Science vol. 674, Springer-Verlag, Berlin, 1993.

[Ezpeleta, 95] EZPELETA, J.; COLOM, J.M.; MARTINEZ, J. A Petri Net Based Deadlock Prevention Policy for Flexible Manufacturing Systems, IEEE Transactions on Robotics and Automation, vol. 11, nº 2, IEEE, 1995.

[Genrich, 81] GENRICH, H.J. e LAUTENBACH, K. System Modeling with High-Level Petri Nets, Theoretical Computer Science, vol. 13, 1981.

[Hasegawa, 84] HASEGAWA, K. et al Proposal of Mark Flow Graph for Discrete System Control, Transactions of SICE, vol. 20, nº 2, Tokyo, 1984 (em japonês).

[Hasegawa, 94] HASEGAWA, K e OHNO, H. Equivalent Transform of C/E System based on S-invariant, Systems Science, vol. 20, 1994.

[Ho, 87] HO, Y.C. Editorial - Basic Research, Manufacturing Automation, and Putting the Cart Before the Horse, IEEE Transactions on Automatic Control, vol. AC-32, nº 12, IEEE, 1987

[Jensen, 90] JENSEN, K. Coloured Petri Nets: A High Level Language for System Design and Analysis, Advances in Petri Nets 1990, Lecture Notes In Computer Science vol. 483, Springer-Verlag, Berlin, 1990.

[Kreyszig, 84] KREYSZIG, E. Matemática Superior, vol.2, LTC Editora, Rio de Janeiro, 1984

[Krückeberg, 87] KRÜCKEBERG, F. e JAXY, M. Mathematical Methods for Calculating Invariants in Petri Nets, Advances in Petri Nets 1987, Lecture Notes In Computer Science vol. 266, Springer-Verlag, Berlin, 1987.

[Lakos, 93] LAKOS, C.A. e KEEN, C.D. Applying Invariant Analysis to Modular Petri Nets, Australian Computer Science Conference, Brisbane, 1993.

[Lautenbach, 87] LAUTENBACH, K. Linear Algebraic Techniques for Place/Transition Nets, Advances in Petri Nets 1986, Lecture Notes In Computer Science vol. 254, Springer-Verlag, Berlin, 1987.

[Levis, 87] LEVIS, A. et al, Challenges to control - A collective view, IEEE Transactions on Automatic Control, vol AC-32, IEEE, 1987.

[Memmi, 87] MEMMI, G. e VAUTHERIN, J. Analyzing Nets by the Invariant Method, Advances in Petri Nets 1986, Lecture Notes In Computer Science vol. 254, Springer-Verlag, Berlin, 1987.

[Miyagi, 88] MIYAGI, P.E. Control Systems Design, Programming and Implementation for Discrete Event Productions Systems by Using Mark Flow Graph, Doctoral Thesis, Tokyo Institute of Technology, Tokyo, 1988.

[Miyagi, 96] MIYAGI, P.E. Controle Programável, Ed. Edgar Blucher Ltda., São Paulo, 1996.

[Murata, 89] MURATA, T. Petri Nets: Properties, Analysis and Applications, Proceedings of the IEEE, vol. 77, no. 4, IEEE, 1989.

[Peterson, 81] PETERSON, J.L. Petri Net Theory and the Modeling of Systems, Prentice-Hall, Englewood Cliffs, 1981.

[Reisig, 85] REISIG, W. Petri Nets, an Introduction, Springer-Verlag, Berlin, 1985.

[Reisig, 92] REISIG, W. Combining Petri Nets and Other Formal Methods, Application and Theory of Petri Nets, Lecture Notes In Computer Science vol. 616, Springer-Verlag, Berlin, 1992.

[Schrijiver, 86] SCHRIJIVER, A. Theory of Linear and Integer Programming, John Wiley & Sons, Chichester, 1986.

[Shimada, 97] SHIMADA, L.M. Estruturação do Problema de Planejamento em uma Abordagem Baseada em IA e no Formalismo de Redes de Petri, Tese de Doutoramento, Escola Politécnica da USP, São Paulo, 1997.

[Silva J.R., 95] SILVA, J.R. e MIYAGI, P.E. PFS/MFG: A High Level Net for the Modeling of Discrete Manufacturing Systems, Balanced Automation Systems - Architectures and Design Methods, IFIP/Chapman&Hall, London, 1995.

[Silva J.R., 96] SILVA, J.R. e MIYAGI, P.E. A Formal Approach to PFS/MFG: A Petri Net Representation of Discrete Manufacturing Systems, Studies in Informatics and Control, Informatics and Control Publ., Bucharest, 1996.

[Silva M., 88] SILVA, M. e COLOM, J.M. On the Computation of Structural Synchronic Invariants in P/T Nets, Advances in Petri Nets 1988, Lecture Notes In Computer Science vol. 340, Springer-Verlag, Berlin, 1988.

[Silva M., 89] SILVA, M. e VALETTE, R. Petri Nets and Flexible Manufacturing, Advances in Petri Nets 1989, Lecture Notes In Computer Science vol. 424, Springer-Verlag, Berlin, 1989.

[Silva M., 91] SILVA, M.; COLOM, J.M. e CAMPOS, J. Linear Algebraic Techniques for the Analysis of Petri Nets, In Recent Advances in Mathematical Theory of Systems, Control, Networks, and Signal Processing, Mita Press, Tokyo, 1991.

[Treves, 89] TREVES, N. A Comparative Study of Different Techniques for Semi-flows Computation in Place/Transition Nets, Advances in Petri Nets 1989, Lecture Notes In Computer Science vol. 424, Springer-Verlag, Berlin, 1989.

[Yamalidou, 96] YAMALIDOU, K. et al. Feedback Control of Petri Nets Based on Place Invariants, in Automatica, vol. 32, n° 1, IFAC, London, 1996.

APÊNDICE

A) Álgebra Linear

A.1 Forma Normal de Hermite

Uma matriz de posto máximo está na forma normal de Hermite [Schrijiver86] se ela tiver a forma [B ¦ 0], onde B é uma matriz não singular, triangular, não-negativa, em que cada linha tem um elemento máximo localizado na diagonal principal de B.

As seguintes operações são chamadas de operações elementares (unimodulares):

(i) troca entre 2 colunas;

(ii) multiplicação de uma coluna por -1;

(iii) adição a uma coluna de um múltiplo inteiro de outra coluna.

Toda matriz racional de posto máximo pode ser transformada na forma normal de Hermite através de uma série de operações elementares sobre suas colunas. Além disso, esta forma normal é única.

A.2 Método de Eliminação de Fourier-Motzkin

Este método foi definido inicialmente para resolver o seguinte sistema de inequações: dada a matriz A e o vetor b, achar as soluções de $A \cdot x \leq b$

Suponha que A tenha m colunas e n colunas. Podemos supor, sem perda da generalidade, que todos os elementos da primeira coluna de A sejam 0, +1 ou -1 (pois podemos multiplicar cada desigualdade por um número escalar positivo. O sistema a ser resolvido é então (rearranjando as inequações):

$x_1 + a_i x' \leq b_i \quad (i = 1, ..., m')$

$-x_1 + a_i x' \leq b_i \quad (i = m' + 1, ..., m'')$

$a_i x' \leq b_i \quad (i = m'' + 1, ..., m)$

onde $x = (x_1, ..., x_n)^T$ e $x' = (x_2, ..., x_n)^T$ e a_i são as linha de A tirando o primeiro elemento. Resolvendo este sistema em x_1, temos:

$a_j x' - b_j \leq b_i - a_i x' \quad (i = 1, ..., m'; j = m' + 1, ..., m'')$

$a_i x' \leq b_i \quad (i = m'' + 1, ..., m)$

Ou seja,

$(a_j + a_i) x' \leq b_i + b_j \quad (i = 1, ..., m'; j = m' + 1, ..., m'')$

$a_i x' \leq b_i \quad (i = m'' + 1, ..., m)$

Repetindo este procedimento, podemos eliminar sucessivamente as primeiras n - 1 componentes de x, e terminamos com um problema equivalente em apenas uma variável, que é trivial. A partir da solução deste sistema final, basta fazer o procedimento contrário para encontrar a solução do problema original.

Este método pode ser utilizado para encontrar as soluções não negativas de um sistema A x = b. Primeiro, transforma-se o sistema A x = b no sistema equivalente $[I \mid A'] . (y \mid z)^T = b'$. A seguir, aplica-se o procedimento acima para o sistema $z \geq 0$, $A'. z \leq b'$.

B) Subclasses de Redes Lugar/Transição

Dependendo das restrições na sua estrutura, uma rede Lugar/Transição ordinária pode ser classificada da seguinte maneira:

i) uma máquina de estado (SM) é uma rede P/T ordinária onde cada transição possui exatamente um lugar de saída e um lugar de entrada. Ou seja:

$| t\bullet | = | \bullet t | = 1$ para todo $t \in T$.(onde | C | indica a cardinalidade do conjunto C)

ii) um grafo marcado (GM) é uma rede P/T ordinária onde cada lugar possui exatamente uma transição de saída e uma transição de entrada. Ou seja:

$| p\bullet | = | \bullet p | = 1$ para todo $p \in P$

iii) uma rede *free-choice* (FC) é uma rede P/T ordinária onde cada arco saindo de um lugar ou é o único arco saindo deste lugar, ou o único arco chegando numa transição. Ou seja:

para todo $p_1, p_2 \in P$, $p_1\bullet \cap p_2\bullet \neq \varnothing \Rightarrow | p_1\bullet | = | p_2\bullet | = 1$

iv) uma rede *free-choice* estendida (EFC) é uma rede P/T ordinária onde:

para todo $p_1, p_2 \in P$, $p_1\bullet \cap p_2\bullet \neq \varnothing \Rightarrow p_1\bullet = p_2\bullet$

C) MFG

Mark Flow Graph (MFG) é um grafo derivado das redes Condição-Evento, visando a modelagem e controle de sistemas discretos de manufatura. A idéia principal é basear o projeto de sistemas de manufatura em macro elementos, tais como estações de montagem, desmontagem, buffers, etc., de forma a contemplar a estrutura básica dos elementos característicos dos sistema e com isso simplificar a tarefa de modelagem.

O MFG é composto pelos seguintes elementos estruturais:

- Box: equivale a uma condição (pode conter no máximo uma marca);
- Transição: equivale a um evento;
- Arco orientado: conecta boxes e transições para indicar a relação entre uma condição e os pré e pós-eventos relacionados a ela. Um arco orientado sempre conecta elementos de tipos distintos. Não existe limite para o número de arcos que saem ou entram dos boxes e transições, mas num par transição-box pode existir no máximo um arco entre eles (o MFG é uma rede pura, portanto).
- Marca (*token*): indica que uma determinada condição (box) é verdadeira.
- Portas (*gates*): representam um tipo especial de relação de fluxo, onde o disparo de uma transição não retira marcas do local de origem do sinal. Uma porta habilitadora (inibidora) habilita (inibe) a ocorrência da transição quando o sinal de origem for "1". Uma porta é externa quando a origem do sinal não faz parte do grafo, ou seja, quando ela indica a entrada de um sinal binário gerado por algum dispositivo externo.

- Arco de sinal de saída: este arco envia um sinal binário do box para os dispositivos externos da rede. Quando houver uma marca no box, o sinal é "1"; caso contrário, é "0".

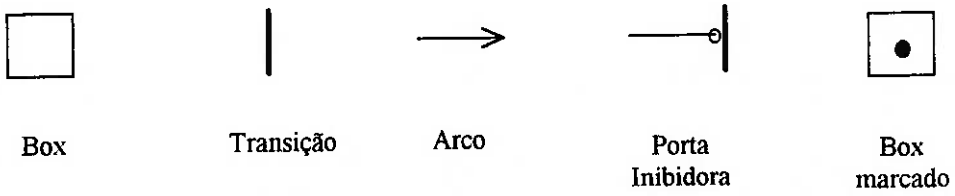


Figura C.1 - Elementos básicos do MFG

O estado de um sistema pode ser representado pelo arranjo das marcas no grafo. Um arranjo das marcas define uma marcação. A marcação inicial é definida pelo arranjo no estado inicial.

O comportamento dinâmico do sistema (alteração dos esatdos) é causado pelo disparo das transições, que seguem as seguintes regras.

Habilitação de Disparo

Uma transição está habilitada para disparo se as seguintes condições estão satisfeitas:

- nenhum box no pré-set da transição está marcado;
- todos os boxes no pós-set da transição estão marcados;
- nenhuma porta habilitadora interna está no estado de desabilitação;
- nenhuma porta inibidora interna está no estado de inibição.

Disapro de Transição

Uma transição é disparável se ela está habilitada e

- nenhuma porta habilitadora externa está no estado de desabilitação;
- nenhuma porta inibidora externa está no estado de inibição.

Se uma transição é disparável, ela dispara imediatamente, exceto quando houver conflitos ou atrasos de tempo que serão discutidos posteriormente.

No disparo, as marcas no interior de todos os boxes no pré-set da transição desaparecem e imediatamente surgem marcas no interior de todos os boxes no pós-set. As marcas nas origens das portas não se alteram. Considera-se ainda que o disparo ocorre num intervalo infinitamente pequeno. No MFG, os disparos são discretos no tempo, existindo uma precedência entre eles. Esta ordem de precedência é denominada seqüência de disparo e a ordenação está baseada nesta seqüência temporal.

Se na marcação inicial não for admitida mais de uma marca no interior de um box, então, pelas regras de disparo, é impossível que um box já marcado receba nova marca. Ou seja, após qualquer sequência de disparo, existe no máximo uma marca em cada box.

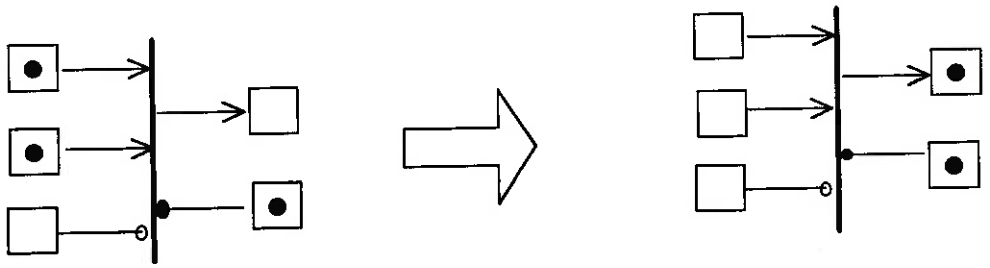


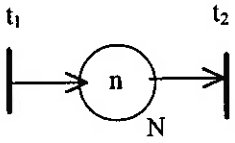
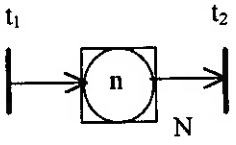
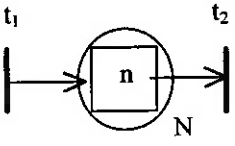
Figura C.2 - Disparo de uma transição

Modularização do MFG

No modelo original, boxes podem aceitar no máximo uma marca. Em casos práticos, a capacidade de alojar mais de uma marca é bastante efetiva para representar

elementos como estoques e magazines.Isto é possível com o uso de alguns módulos básicos, isto é, macro-boxes com capacidade de manipular várias marcas de uma só vez.

O box *capacidade* tem um símbolo N associado que indica a sua capacidade de aceitar N marcas. O box *agrupador* tem uma função similar a de uma montagem, onde N marcas entram e apenas uma sai (ou seja, N itens entram para serem montados e um item só sai). O box *dispersor* tem uma função similar a de uma montagem, onde uma marca entra e N marcas saem (um item entra para ser desmontado e N itens saem).

	Box capacidade	Box agrupador	Box dispersor
N = capacidade total das marcas n = número de marcas presente			
condição necessária para o disparo de t_1	n < N	n < N	n = 0
condição necessária para o disparo de t_2	n > 0	n = N	n > 0

É importante observar que estes macro-elementos poderiam ser representados usando apenas boxes e transições, como por exemplo na figura C.3.

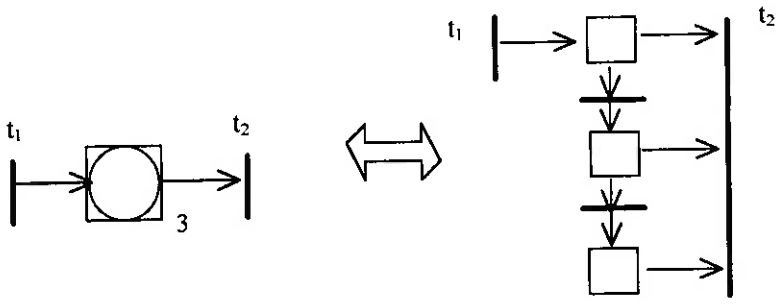


Figura C.3 - Equivalência do macro-elemento agrupador com capacidade 3

D) Listagens dos programas em MAPLE

D.1 Entrada das Matrizes

```
A := array(1..6,1..7,[
[ 1,  0,  0,  0, -1,  0, 0],
[-1,  1,  0,  0,  0,  0, 0],
[ 0, -1,  0,  1,  0,  0, 0],
[-1,  0,  1,  0,  0,  0, 0],
[ 0,  0, -1,  1,  0,  0, 0],
[ 0,  0,  0, -1,  1,  0, 0]]):

A1 := array(1..4,1..3,[
[ 1,  0,  0],
[ 0, -1, -1],
[-1,  1,  0],
[-1,  0,  1]]):
```

D.2 Funções para Calcular e Compor os Invariantes

```
#######################################################
### Rotina para encontrar os invariantes mínimos
### Maple V release 3
### Song San Woei 29/07/1997
###
### Parametro de Entrada:
###    Incid - Matriz de incidencia
###       tamanho: m lugares X n transições (S-invar) ou
###                n trans   X m lugares    (T-invar)
###
### Saída:
###    matriz onde cada linha é um invariante mínimo
###
#######################################################

Invariante := proc(Incid)
        local m,n,Linha_Zeros,Un,Up,U1,U0,X,i,j,k,kk;
        option remember;

m := rowdim(Incid); n:=coldim(Incid);
Linha_Zeros := array(sparse, 1..1,1..m+n):
U1 := stack(Linha_Zeros, augment(Incid, array(identity, 1..m,1..m))):
for i to n do
  U0 := op(U1);
  U1 := op(Linha_Zeros):
  Un := op(Linha_Zeros):
  Up := op(Linha_Zeros):
  m := rowdim(U0);
```

```
 for j from 2 to m do
  if U0[j,i] = 0 then     U1 := stack(U1, row(U0,j))
    elif U0[j,i] <0 then  Un := stack(Un, row(U0,j))
    else Up := stack(Up, row(U0,j))
  fi
 od:
  for k from 2 to rowdim(Un) do
     for kk from 2 to rowdim(Up) do
       X := abs(Up[kk,i])*row(Un, k) + abs(Un[k,i])*row(Up, kk):
       U1 := stack(U1, X)
     od
   od

od:
if rowdim(U1) > 1 then
   U1:=submatrix(U1, 2..rowdim(U1),n+1..coldim(U1)):
   for i to rowdim(U1) do
      mdc:=gcd(U1[i,1],U1[i,2]):
      for ind from 3 to coldim(U1) do mdc:=gcd(mdc,U1[i,ind]): od:
      if mdc > 1 then U1 := mulrow(U1,i,1/mdc) fi:
   od:
  else
   U1:=submatrix(Linha_Zeros, 1..1,n+1..coldim(Linha_Zeros))
fi:
RETURN(op(U1)):
end:

#####################################################
### Rotina para compor os S-invariantes mínimos
### Maple V release 3
### Song San Woei 15/08/1997
###
### Parametro de Entrada:
###    X1 - Matriz contendo os S-invar do nível superior
###    Pos1 - indice do box_temporizado que esta sendo expandido
###    X2 - Matriz contendo os S-invar na subrede
###    (Obs: a primeira coluna de X2 deve corresponder ao box de tempo)
###
### Saida:
###    matriz onde cada linha é um invariante mínimo
###
#####################################################

 S_Compor := proc(X1,Pos1,X2)
          local D,U1,X,i,j,m,mdc,ind,AUX1,AUX2;
          option remember;

 if Pos1 >= coldim(X1) or Pos1 < 1
```

```
  then ERROR(`Indice incorreto`)
fi:

U1 := extend(X1,0,coldim(X2)-1,0):
D := array(sparse, 1..1,1..coldim(X1)):
for i from 1 to rowdim(X1) do
  if X1[i,Pos1] > 0
     then D := stack(D, row(X1,i))
  fi
od:
m := rowdim(D):

for i from 1 to rowdim(X2) do
  if X2[i,1] = 0
    then X:=augment(array(sparse, 1..1,1..coldim(X1)), submatrix(X2, i..i,
2..coldim(X2))):
       U1:=stack(U1, X):
    else for j from 2 to m do
       AUX1:=submatrix(D, j..j, 1..coldim(D)):
       AUX2:=submatrix(X2, i..i, 2..coldim(X2)):
       AUX1:=mulrow(AUX1,1,X2[i,1]):
       AUX2:=mulrow(AUX2,1,D[j,Pos1]):
       X:=augment(AUX1, AUX2):
       X[1,Pos1]:=0:
       mdc:=gcd(X[1,1],X[1,2]):
       for ind from 3 to coldim(X) do mdc:=gcd(mdc,X[1,ind]): od:
       if mdc > 1 then X := mulrow(X,1,1/mdc) fi:
       U1:=stack(U1, X):
      od:
  fi
od:
RETURN(op(U1))
end:

#####################################################
### Rotina para compor os T-invariantes mínimos
### Maple V release 3
### Song San Woei 15/08/1997
###
### Parametro de Entrada:
###   Y1 - Matriz contendo os T-invar do nível superior
###   Pos1 - indice da atividade de entrada
###   Y2 - Matriz contendo os T-invar na subrede
###   (Obs: as colunas 1 e 2 de Y2 devem corresponder as atividades
###         de entrada e saida, respectivamente)
###
### Saída:
###   matriz onde cada linha é um invariante mínimo
```

```
###
#################################################

T_Compor := proc(Y1,Pos1,Y2)
          local D,U1,X,i,j,m,mdc,ind;
          option remember;

if Pos1 >= coldim(Y1) or Pos1 < 1
  then ERROR(`Indice incorreto`)
fi:

D := array(sparse, 1..1,1..coldim(Y1)):
U1 := array(sparse, 1..1,1..coldim(Y1)+coldim(Y2)-2):
for i from 1 to rowdim(Y1) do
  if Y1[i,Pos1] > 0
     then D := stack(D, row(Y1,i)):
     else X:=augment(submatrix(Y1, i..i, 2..coldim(Y1)), array(sparse,
1..1,1..coldim(Y2)-2)):
       U1:=stack(U1, X):
  fi
od:
m := rowdim(D):

for i from 1 to rowdim(Y2) do
  if Y2[i,1] = 0
    then X:=augment(array(sparse, 1..1,1..coldim(Y1)), submatrix(Y2, i..i,
3..coldim(Y2))):
       U1:=stack(U1, X):
    else for j from 2 to m do
      X:=augment(Y2[i,1]*submatrix(D, j..j, 1..coldim(D)), D[j,Pos1]*submatrix(Y2,
i..i, 3..coldim(Y2))):
       mdc:=gcd(X[1,1],X[1,2]):
       for ind from 3 to coldim(X) do mdc:=gcd(mdc,X[1,ind]): od:
       if mdc > 1 then X := mulrow(X,1,1/mdc) fi:
       U1:=stack(U1, X):
      od:
  fi
od:
if rowdim(U1) > 1 then
  RETURN(submatrix(U1, 2..rowdim(U1),1..coldim(U1)))
 else
  RETURN(submatrix(U1, 1..1,1..coldim(U1)))
fi:
end:
```

D.3 Programa Principal

```
restart:
with(linalg):
read `/maplev3/entrada.txt`:
read `/maplev3/invar.txt`:
XX := Invariante(transpose(A)):
YY := Invariante(A):

### repetir para cada atividade
read `/maplev3/entrada2.txt`:
Coluna:=array(sparse,1..rowdim(A1),1..1):
Coluna[1,1]:=-1: Coluna[2,1]:=1:
C:=augment(Coluna,A1):
XX1:=Invariante(transpose(C)):
YY1:=Invariante(A1):
### Alterar próxima linha conforme o índice da atividade
Box_Tempo:=3: Ativ_Entrada:=3:
XX:=S_Compor(XX,Box_Tempo,XX1):
YY:=T_Compor(YY,Ativ_Entrada, YY1):
```