Rede de Petri e o aplicativo Platform Independent Petri net Editor (PIPE)

Caio Cesar Fattori; Célia Hanako Kano, Fabrício Junqueira, Paulo Eigi Miyagi

Escola Politécnica da Universidade de São Paulo

30 de julho de 2012

Lista de Figuras

Sumário

Capítulo 1

Introdução

O objetivo deste material é complementar obra Miyagi (1996) com a formalização da Rede de Petri (RdP) e a apresentação de um aplicativo de modelagem e análise de rede de Petri conhecida como PIPE. O aplicativo PIPE é baseado em Java e pode ser encontrado para download na página de seus desenvolvedores http://pipe2.sourceforge.net/.

Para explicar o aplicativo é necessário primeiro apresentar a definição adotada pelo PIPE para a rede de Petri, seus elementos e suas propriedades.

1.1 Rede de Petri (RdP)

A rede de Petri (RdP), como uma técnica de representação gráfica e matemática, provê um ambiente uniforme para modelagem, análise formal e projeto de sistemas a eventos discretos (SEDs) (Adam et al., 1998, Nassar et al., 2008, Zurawski and Zhou, 1994). Ela é muito efetiva como técnica de descrição e especificação de processos (Morales et al., 2007, Yoo et al., 2010, Hamadi and Benatallah, 2003). A rede de Petri é uma representação que pode ser usada tanto no nível conceitual quanto no nível funcional, em que o sistema ou processo podem ser analisados e validados antes de prosseguir com o projeto detalhado e a implementação. A RdP é uma ferramenta de comunicação entre pessoas relacionadas ao projeto, permitindo uma fácil interpretação e nítida identificação dos estados e ações. Possui a vantagem de um mesmo modelo poder ser usado para análise das propriedades comportamentais e para avaliação de desempenho, assim como para a especificação da solução de controle. Pode ser usada para identificar propriedades de sistemas como sincronização de processos, eventos assíncronos, operações concorrentes, conflitos ou compartilhamento de recursos, etc. Matematicamente, a RdP pode ser descrita como um conjunto de equações algébricas lineares e, por isso, pode ser usada para a verificação formal de relações de precedência entre eventos e condições.

Desde sua apresentação por Carl Adam Petri (Brauer and Reisig, 2006), a RdP tem sido usada na modelagem e análise de diferentes tipos de sistemas e aplicações tais como: protocolos distribuídos (Kaneshiro et al., 2008), aplicações industriais (Zurawski and Zhou, 1994, Nassar et al., 2008), tratamento de falhas (Morales et al., 2007), fluxo de processos (Kiepuszewski et al., 2003), controle supervisório (Garcia Melo et al., 2008, Lee et al., 2005), entre outras.

1.1.1 Formalização

Inicialmente tem-se a REDE DE PETRI CONDIÇÃO-EVENTO (*C/E net*) que matematicamente, é descrita como uma 4-tupla

$PN_{CE} = (P, T, F, M)$

em que:

P é o conjunto de elementos passivos chamados de LUGARES,

T é o conjunto de elementos ativos chamados de TRANSIÇÕES,

F_{CE} é o conjunto de relacionamentos entre os elementos passivos e os elementos ativos chamados de ARCOS ORIENTADOS,

M_{CE} é um vetor de recursos chamados de MARCAÇÃO, isto é, um vetor de números binários em que cada elemento m_i indica o número de MARCAS no LUGAR p_i, isto é $m_i = M_{CE}(p_i)$ e, $\forall p \in P, M_{CE}(p) \in (0, 1)$,e

onde,

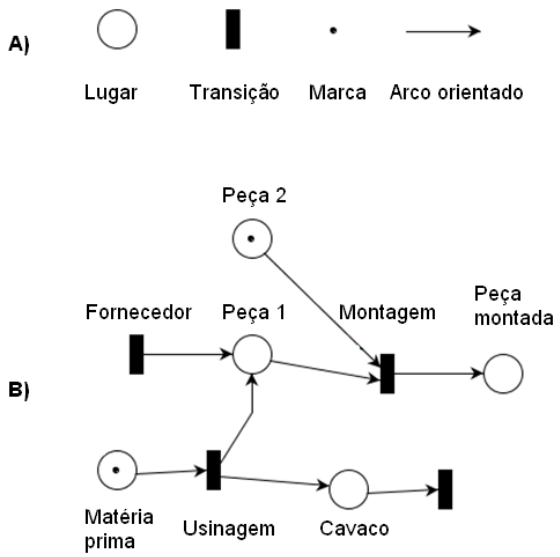


Figura 1.1: A) Representação gráfica dos elementos da RdP; B) Exemplo de RdP como modelo de um processo.

P e T são conjuntos finitos, não nulos ($P \cup T \neq \emptyset$) e disjuntos ($P \cap T = \emptyset$), e
$F_{CE} \subseteq (P \times T) \cup (T \times P)$.

A representação gráfica dos LUGARES da RdP são circunferências; a representação gráfica das TRANSIÇÕES são retângulos; a representação gráfica dos ARCOS ORIENTADOS são setas apontando do elemento de origem para o elemento de destino; e a representação gráfica das MARCAS são pontos dentro dos LUGARES, assim como na Figura 1.1.

Os estados do sistema ou processo são representados pela distribuição das MARCAS nos LUGARES da rede. A dinâmica da RdP depende da alteração do estado da RdP, resultante do disparo das TRANSIÇÕES. Pode-se assim adotar a idéia de que as MARCAS fluem pela rede direcionadas pelos ARCOS ORIENTADOS, isto é, fluem de certos LUGARES para outros LUGARES passando pelas TRANSIÇÕES disparadas.

- Para que uma MARCA flua de um LUGAR $p \in P$ para uma TRANSIÇÃO $t \in T$ por uma ARCO ORIENTADO $f \in F_{CE}$, esse LUGAR deve possuir uma MARCA, ou seja, $M_{CE}(p) = 1$. Esse LUGAR p é dito então uma pré-condição da TRANSIÇÃO t e sua representação é $p \in^{\bullet} t$, em que ${}^{\bullet}t$ é o conjunto de pré-condições da TRANSIÇÃO t.

- Para que uma MARCA flua de uma TRANSIÇÃO $t \in T$ para um LUGAR $p \in P$ por um ARCO ORIENTADO $f \in F_{CE}$, esse LUGAR não deve possuir MARCAS, ou seja, $M_{CE}(p) = 0$. Esse LUGAR p é dito então uma pós-condição da transição t e sua representação é $p \in t^{\bullet}$, em que $t^{\bullet}$ é o conjunto de pós-condições da TRANSIÇÃO t.

Quando uma TRANSIÇÃO possui todas as suas pré-condições e pós-condições atendidas essa TRANSIÇÃO é dita habilitada para o disparo. Uma TRANSIÇÃO habilitada pode disparar e esse instante é denominado passo. Quando t dispara:

- todo LUGAR p_i tal que $p_i \in^{\bullet} t$ tem sua MARCAÇÃO alterada entre o passo k e o passo $k+1$ da seguinte maneira $M_{CE_{k+1}}(p_i) = 0$, e

- todo LUGAR p_j tal que $p_j \in t^{\bullet}$ tem sua MARCAÇÃO alterada entre o passo k e o passo $k+1$ da seguinte maneira $M_{CE_{k+1}}(p_j) = 1$.

Inclusão de novos conceitos - LUGAR com capacidade e ARCOS múltiplos Com o aumento do uso da RdP, alguns conceitos foram revisados com o objetivo de simplificar a representação do modelo. Um destes casos envolve a atribuição CAPACIDADE DE MARCAS $C(p)$ para os LUGARES em que $\forall p \in P, C(p) \in \mathbb{N}^{\star}$, que simplifica a representação de um conjunto de LUGARES em um único (macro) LUGAR, com uma nova definição do vetor de MARCAS M para $\forall p \in P, M(p) \in \mathbb{N}$, representado graficamente com um número natural próximo ao LUGAR, assim como na Figura 1.2. Outro importante

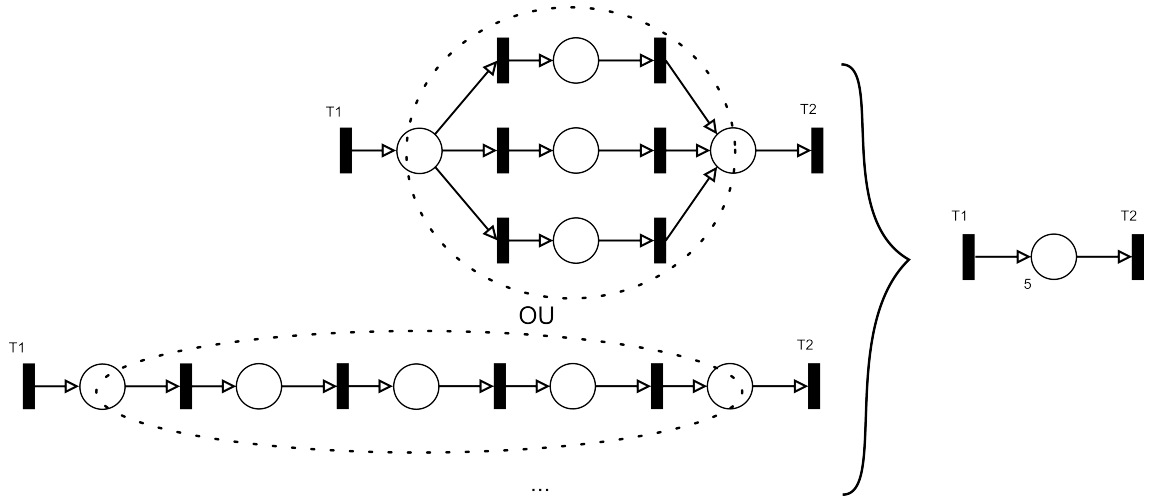


Figura 1.2: Complemento de capacidade dos lugares

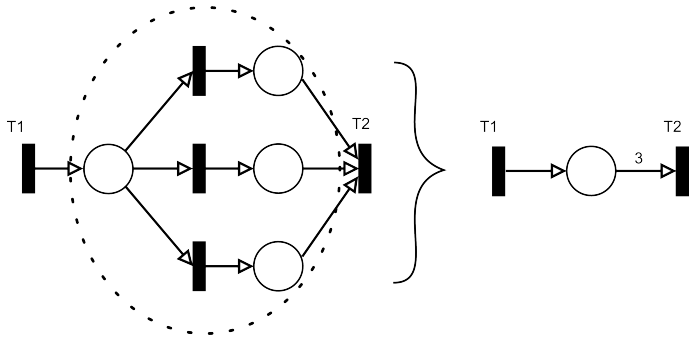


Figura 1.3: Complemento de peso dos arcos orientados

conceito é o de PESO DOS ARCOS $W(f)$ que, analogamente à capacidade dos LUGARES, indica a existência de múltiplos ARCOS orientados entre um LUGAR e uma TRANSIÇÃO ou entre uma TRANSIÇÃO e um LUGAR, representado graficamente com um número natural próximo ao ARCO orientado, assim como na Figura 1.3. Matematicamente, tem-se assim que $\forall f \in F_{LT}, W(f) \in \mathbb{N}^{\star}$, que também pode ser representado como $W(x,y) = \mathbb{N} \mid (x \in P, y \in T) \cup (x \in T, y \in P)$. Esta rede é chamada de REDE DE PETRI LUGAR/TRANSIÇÃO (P/T *net*) em que $PN_{LT} = (P, T, F, W, M)$.

Com isso, na RdP LUGAR/TRANSIÇÃO, para que MARCAS fluam de um LUGAR p para uma TRANSIÇÃO t por uma ARCO orientado f, o LUGAR deve possuir, no mínimo, o número de MARCAS determinado no PESO do ARCO orientado, ou seja, $M(p) \geq W(f)$. Também, para que MARCAS fluam de uma TRANSIÇÃO t para um LUGAR p por um ARCO orientado f, o LUGAR deve possuir CAPACIDADE de receber o número de MARCAS determinado no PESO do ARCO orientado, ou seja, $C(p) - M(p) \geq W(f)$. Assim como na RdP CONDIÇÃO EVENTO, os LUGARES a partir dos quais fluem as MARCAS por ARCOS orientados para uma TRANSIÇÃO integram o conjunto de pré-condições da TRANSIÇÃO e os LUGARES que recebem as MARCAS de uma TRANSIÇÃO integram o conjunto de pós-condições desta TRANSIÇÃO.

Quando uma TRANSIÇÃO possui todas as suas pré-condições e pós-condições atendidas, essa TRANSIÇÃO é dita habilitada para o disparo. Uma TRANSIÇÃO habilitada pode disparar e esse instante é denominado passo). Quando t dispara:

- todo lugar p_i tal que $p_i \in^{\bullet} t$ tem sua MARCAÇÃO alterada entre o passo k e o passo $k+1$ da seguinte maneira $M_{k+1}(p_i) = M_k(p_i) - W(p_i, t)$, e
- todo lugar p_j tal que $p_j \in t^{\bullet}$ tem sua MARCAÇÃO alterada entre o passo k e o passo $k+1$ da seguinte maneira $M_{k+1}(p_j) = M_k(p_j) + W(t, p_j)$.

Extensão da RdP - ARCO INIBIDOR Para ampliar o poder de modelagem da RdP foram posteriormente propostas extensões para acrescentar novas funcionalidades e para simplificar o modelo. Uma extensão que simplifica o modelo em RdP é o ARCO INIBIDOR que envolve o conceito de rede dual. A

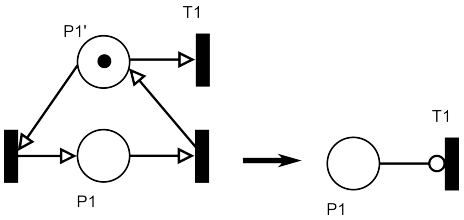


Figura 1.4: Complemento de arco orientado inibidor

Figura 1.5: Representação gráfica da transição temporizada na RdP T-temporizada

rede dual é derivada em correspondência a cada LUGAR da RdP de forma que se p'_1 é o dual de p_1, com $p_1 \in P$, em que se $M(p_1) = 0$, então no caso de RdP C/E $M(p'_1) = 1$, e se $M(p_1) \neq 0$, então $M(p'_1) = 0$. Assim, ao conectar o LUGAR dual à uma TRANSIÇÃO, esta pode ser entendida como o LUGAR da rede que inibe o disparo da TRANSIÇÃO. O ARCO INIBIDOR é representado graficamente como uma linha com uma circunferência na ponta mais próxima a TRANSIÇÃO, assim como na Figura 1.4.

Os ARCOS INIBIDORES também são pré-condições para uma TRANSIÇÃO. Se um ARCO INIBIDOR f_I relaciona o LUGAR p a TRANSIÇÃO t, este ARCO determina que a pré-condição da TRANSIÇÃO está atendida apenas na condição de MARCAÇÃO nula, ou seja, $M(p) = 0$. Porém, diferentemente dos ARCOS ORIENTADOS normais, os ARCOS INIBIDORES não afetam as MARCAS de seus LUGARES.

Extensão da RdP - temporizações A RdP também é usada para resolver problemas de agendamento em sistemas dinâmicos, mas para isso foi feita a inclusão de um elemento que incorpora o conceito de tempo. Assim surgiram as RdP temporizadas, que podem ser de dois tipos: P-temporizadas ou T-temporizadas. Nessas RdP, o modelo inclui também um contador de tempo que é baseada na idéia de ciclo de tempo. Um ciclo de tempo é estabelecido quando nenhuma TRANSIÇÃO $t \in T$ pode mais disparar. A RdP P-temporizada inclue aos LUGARES da RdP uma propriedade que define quantos ciclos de tempo uma MARCA deve permanecer dentro de um LUGAR antes de poder ser retirada devido a um disparo de TRANSIÇÃO. A RdP T-temporizada inclue nas TRANSIÇÕES uma propriedade que define por quantos ciclos de tempo as pré-condições e pós-condições devem ser atendidas para que a TRANSIÇÃO dispare. Na RdP T-temporizada, caso uma das pré-condições ou pós-condições não esteja mais atendida durante a contagem de ciclos de tempo, o contador de ciclo de tempo dessa TRANSIÇÃO é zerado. Se a TRANSIÇÃO disparar, seu contador de ciclo de tempos também é zerado. A representação gráfica da transição temporizada na RdP T-temporizada é um retângulo oco, como na Figura 1.5.

Tipo de RdP que pode ser usada no PIPE A RdP implementada no aplicativo PIPE é uma RdP LUGAR/TRANSIÇÃO T-temporizada com ARCOS INIBIDORES e, é uma 6-tupla $PN = (P, T_N, T_T, F_N, F_I, M_0)$,

P é o conjunto de LUGARES,

T_N é o conjunto de TRANSIÇÕES não temporizadas,

T_T é o conjunto de TRANSIÇÕES temporizadas,

F_N é o conjunto dos ARCOS ORIENTADOS normais,

F_I é o conjunto dos ARCOS INIBIDORES,

M_0 é um vetor onde cada elemento que representa a quantidade inicial de MARCAS no LUGAR é também conhecido como MARCAÇÃO INICIAL dos LUGARES.

Para cada LUGAR p é necessário definir sua capacidade $C(p)$.

Nas TRANSIÇÕES temporizadas t_T é necessário definir quantos ciclos de tempo uma TRANSIÇÃO temporizada deve esperar para o disparo após estar habilitada.

Para cada ARCO ORIENTADO f_N é necessário definir o PESO DO ARCO $W(f_N)$.

1.1.2 Características dos SEDs

O comportamento dinâmico de SEDs pode ser entendido pela evolução de seus processos que possuem características específicas e que uma ferramenta de modelagem de SEDs deve ser capaz de representar.

Estes processos por sua vez são compostos por um conjunto de atividades. Algumas das características mais importantes dos processos em SEDs são:

1.1.2.1 Sequência

A sequência é a característica que indica uma ordem de realização das atividades envolvidas, isto é, qual conjunto de atividades deve ser concluido para que outro seja iniciado.

Em RdP, atividades são representados por TRANSIÇÕES. Assim, a relação de sequência de duas atividades t_1 e t_2, pode ser representada adicionando-se um LUGAR p_1 e ARCOS orientados de modo que $p_1 \in^\bullet t_2$ e $p_1 \in t_1^\bullet$. Assim, para que a TRANSIÇÃO t_2 seja HABILITADA, é necessário que a TRANSIÇÃO t_1 tenha sido anteriormente disparada. A Figura 1.6 mostra duas atividades numa relação de sequência.

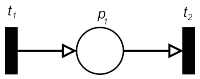


Figura 1.6: Representação em RdP de duas atividades em sequência

1.1.2.2 Paralelismo

O paralelismo é uma característica das atividades que podem ser executados ao mesmo tempo sem que haja interferência da execução de um em outro, mas ambos com a mesma origem, isto é, uma única condição (estado) inicial.

Em RdP, o paralelismo de duas atividades pode ser modelado por duas TRANSIÇÕES t_2 e t_3 e dois LUGARES p_1 e p_2 e ARCOS orientados de modo que $p_1 \in^\bullet t_2$, $p_2 \in^\bullet t_3$. Para representar a mesma condição inicial, adiciona-se mais uma TRANSIÇÃO t_1 e ARCOS orientados para que, $p_1 \in t_1^\bullet$ e $p_2 \in t_1^\bullet$. Assim, quando ocorre o DISPARO da TRANSIÇÃO t_1, tanto a TRANSIÇÃO t_2 quanto a TRANSIÇÃO t_3 ficam habilitadas, mas o disparo de um independe da outra. A Figura 1.7 mostra um paralelismo de duas atividades.

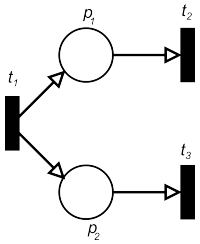


Figura 1.7: Representação do paralelismo em RdP de duas atividades com a mesma condição inicial

1.1.2.3 Sincronização

A sincronização é uma característica de sistemas em que uma atividade depende da execução de outras atividades. Em geral, a sincronização é feita entre atividades em paralelo, mas também pode ocorrer entre duas atividades com origens (processos) totalmente independentes.

Em RdP, a sincronização de duas atividades pode ser modelada por três TRANSIÇÕES t_1,t_2 e t_3 e dois LUGARES p_1 e p_2 e ARCOS orientados de modo que $p_1 \in^\bullet t_3$, $p_2 \in^\bullet t_3$, $p_1 \in t_1^\bullet$ e $p_2 \in t_2^\bullet$. Assim, é necessário que tanto a TRANSIÇÃO t_1 como a TRANSIÇÃO t_2 sejam anteriormente disparadas para que a TRANSIÇÃO t_3 seja habilitada. A Figura 1.8 mostra uma sincronização de duas atividades.

1.1.2.4 Conflito

O conflito é uma característica de sistemas associada a escolha. Quando as atividades estão em conflito, significa que apenas uma delas pode ser executada e a execução de uma dessas atividades inibe a execução das outras atividades que estavam em conflito.

Em RdP, o conflito de duas atividades pode ser modelada por três TRANSIÇÕES t_1,t_2 e t_3 e um LUGAR p_1 em que $p_1 \in^\bullet t_2$, $p_1 \in^\bullet t_3$ e $p_1 \in t_1^\bullet$. Assim, após o DISPARO da TRANSIÇÃO t_1, tanto a TRANSIÇÃO t_2 quanto a TRANSIÇÃO t_3 ficam habilitadas, mas apenas uma delas poderá ser DISPARADA. A Figura 1.9 ilustra este caso.

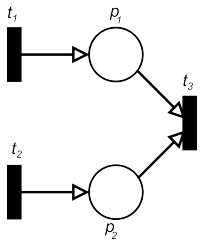


Figura 1.8: Representação em RdP da sicronização de duas atividades.

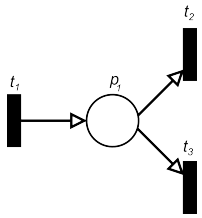


Figura 1.9: Representação em RdP do conflito entre duas atividades

1.1.2.5 Compartilhamento de recursos

O compartilhamento de recursos é uma característica de sistemas em que atividades diferentes dependem de um mesmo recurso para ser executados. Um recurso pode ser um equipamento usado em diferentes atividades em um sistema produtivo, uma informação na memória de um sistema operacional, uma linha de transmissão em um sistema de comunicação, uma máquina ou ferramenta usado em sistemas de manufatura, entre outros das diferentes áreas de SEDs.

Em RdP, o compartilhamento de recursos de duas atividades pode ser modelado por quatro TRANSIÇÕES t_1,t_2, t_3 e t_4 e três LUGARES p_1, p_2 e p_3 em que $p_1 \in^\bullet t_1$, $p_1 \in^\bullet t_2$, $p_2 \in^\bullet t_3$, $p_3 \in^\bullet t_4$, $p_2 \in t_1^\bullet$, $p_3 \in t_2^\bullet$, $p_1 \in t_3^\bullet$ e $p_1 \in t_4^\bullet$ e $M(p_1) = 1$. Assim, tanto a sequência t_1, t_3 quanto a sequência t_2, t_4 dependem de um recurso presente em p_1, mas por ser um recurso único, apenas uma das sequências pode ocorrer de cada vez, mas após essa sequência ocorrer, o recurso volta a ficar disponível. A Figura 1.10 mostra um compartilhamento de recurso de dois processos.

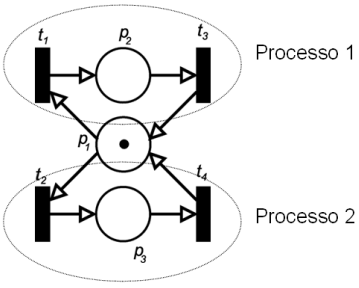


Figura 1.10: Representação em RdP do compartilhamento de recursos entre atividades de dois processos distintos.

1.1.3 Propriedades

Segue as principais propriedades que podem caracterizar o comportamento dinâmico de uma RdP. As definições foram baseadas no trabalho de Murata (1989).

1.1.3.1 Alcançabilidade

A identificação dos estados alcançáveis de um sistema é fundamental para o estudo das propriedades dinâmicas de qualquer sistema. O DISPARO de uma TRANSIÇÃO habilitada muda a distribuição das MARCAS (ou MARCAÇÃO) da RdP. Uma sequência de DISPAROS resulta em uma sequência de MARCAÇÕES. Uma MARCAÇÃO M_n é dita alcançável a partir de uma MARCAÇÃO M_0 se existir uma sequência de

DISPAROS que transforma M_0 em M_n. Uma sequência de DISPAROS é denotada por $\sigma = M_0\, t_1\, M_1\, t_2\, M_2 \cdots t_n\, M_n$ ou simplesmente por $\sigma = t_1\, t_2 \cdots t_n$. Neste caso, M_n é alcançável a partir de M_0 por σ e é representado por $M_0[\sigma > M_n$. O conjunto de todas as possíveis MARCAÇÕES alcançáveis a partir de M_0 em uma rede $(P, T_N, T_T, F_N, F_I, M_0)$ é denotado por $R(P, T_N, T_T, F_N, F_I, M_0)$ ou simplesmente de $R(M_0)$.

O problema da alcançabilidade em RdP é saber se $M_n \in R(M_0)$ para uma dada MARCAÇÃO M_n. Em algumas aplicações, pode ser interessante fazer o estudo de MARCAÇÕES em um subconjunto específico de LUGARES e não levar em conta o restante da rede.

1.1.3.2 Limitabilidade e Segurança

A RdP $(P, T_N, T_T, F_N, F_I, M_0)$ é dita k-limitada ou simplesmente limitada se o número de MARCAS em cada LUGAR da rede não exceder um número finito k para qualquer MARCAÇÃO alcançável a partir de M_0, ou seja, $\forall p \in P | M(p) \leq k$. Uma RdP é dita segura se esta for 1-limitada. Na prática, os LUGARES da RdP são frequentemente usados para representar *buffers* e/ou registradores de dados, itens ou materiais. Assim, assegurar que a rede que modela o sistema é limitada ou segura, é uma forma de garantir que não haverá *overflow* nos *buffers* ou registradores, não importando qual sequência de eventos, isto é, disparos ocorra.

1.1.3.3 Vivacidade

A vivacidade está intimamente relacionada à ausência de *deadlocks* (situações de autotravamento) na operação de um sistema. Uma RdP é dita viva se, independentemente de qual MARCAÇÃO é alcançada a partir de M_0, é sempre possível disparar qualquer TRANSIÇÃO da rede por alguma sequência de DISPARO. Isto significa que a RdP viva garante a operação com ausência de *deadlocks*, não importa qual sequência de DISPAROS foi seguida.

Vivacidade é a propriedade fundamental para a operação regular de muitos sistemas. Entretanto, a identificação desta propriedade nem sempre é simples e o custo de de verificação para alguns sistemas pode ser muito elevado. É possível, então, relaxar a condição de vivacidade e definir diferentes níveis de vivacidade.

Neste sentido, uma TRANSIÇÃO t em uma RdP é dita:

- morta ou L0-viva se nunca pode ser DISPARADA em qualquer sequência de DISPAROS a partir de M_0;
- L1-viva (ou potencialmente disparável) se t pode ser DISPARADA ao menos uma vez em alguma sequência de DISPAROS a partir de M_0;
- L2-viva se, dado um número inteiro positivo k, t pode ser DISPARADO pelo menos k vezes em alguma sequência de DISPAROS a partir de M_0;
- L3-viva se t aparece infinitamente, em alguma sequência de DISPAROS a partir de M_0;
- L4-viva ou viva se t é L1-viva para qualquer MARCAÇÃO M em $R(M_0)$.

A RdP é dita Lk-viva se todas as transições na rede forem Lk-vivas, $k = 0, 1, 2, 3, 4$. A vivacidade L4 é a mais abrangene dos casos e corresponde a definição de vivacidade apresentada inicialmente. É possível verificar que vivacidade L4 implica em vivacidade L3 que implica em vivacidade L2 que implica em vivacidade L1.

1.1.3.4 Reversibilidade

Uma RdP é dita reversível se, para qualquer MARCAÇÃO M em $R(M_0)$, M_0 é alcançável a partir de M. Assim, em uma rede reversível é sempre possível se retornar para a MARCAÇÃO inicial ou estado inicial. Em muitas aplicações, não é tão necessário retornar ao estado inicial quanto retornar a um estado desejado. Portanto, a condição de reversibilidade pode ser relaxada pela definição do estado desejado. Uma MARCAÇÃO M' é considerada um estado desejado se, para cada MARCAÇÃO M em $R(M_0)$, M' é alcançável a partir de M.

1.1.3.5 Cobertura

Uma MARCAÇÃO M em uma RdP é coberta se existe uma MARCAÇÃO M' em $R(M_0)$ tal que $M'(p) \geq M(p)$ para cada p na rede. Cobertura é fortemente relacionada a vivacidade L1. Se M for a MARCAÇÃO mínima para habilitar a TRANSIÇÃO t para o seu DISPARO, então t é morta se e somente se$M$ não estiver coberta. Ou seja, t é L1-viva se e somente se M está coberta.

1.1.3.6 Persistência

Uma RdP é dita persistente se, para quaisquer duas TRANSIÇÕES habilitadas, o DISPARO de uma delas não desabilita a outra. Uma TRANSIÇÃO em uma rede persistente, uma vez habilitada, permanece habilitada até que DISPARE. A noção de persistência é útil no contexto de sistemas assíncronos. A persistência é fortemente relacionada a uma rede sem conflitos.

1.1.3.7 Distância síncrona

A distância síncrona é uma métrica fortemente relacionada ao grau de dependência mútua entre duas TRANSIÇÕES em uma RdP. A distância síncrona entre duas TRANSIÇÕES t_1 e t_2 é definida na RdP como $d_{1\,2} = \underset{\sigma}{max}|\bar{\sigma}(t_1) - \bar{\sigma}(t_2)|$, em que σ é uma sequência de DISPAROS começando em qualquer MARCAÇÃO M em $R(M_0)$ e $\bar{\sigma}(t_i)$ é o número de vezes que a TRANSIÇÃO t_i, com $i = 1, 2$, DISPARA em σ.

1.1.3.8 Equidade

Diferentes noções de equidade foram propostas na literatura de RdP. Aqui serão aborados dois conceitos de equidade: equidade limitada e equidade incondicional (ou global). Duas TRANSIÇÕES t_1 e t_2 possuem uma relação igual-limitada se o máximo de vezes que uma DISPARAR enquanto a outra não DISPARAR é limitada. Uma RdP é dita igual-limitada se cada par de TRANSIÇÕES desta mantem uma relação igual-limitada. Uma sequência de DISPAROS σ é dita incondicionalmente igual se esta é finita ou todas as TRANSIÇÕES da rede aparecem infinitamente em σ. Uma RdP é dita incondicionalmente igual se todas as sequências de DISPAROS σ a partir de M em $R(M_0)$ é incondicionalmente igual.

Capítulo 2

PIPE

Este capítulo tem o objetivo de mostrar passo-a-passo o uso do programa aplicativo PIPE para a modelagem, simulação e análise de Redes de Petri (RdP). O aplicativo PIPE (*Plataform Independent Petri net Editor*) foi inicialmente criado como um projeto de um grupo de alunos de pós-graduação do Departamento de Computação do *Imperial College London*, em 2003. O objetivo era criar um aplicativo para edição e análise de RdPs. Anteriormente, o *Imperial College London* já tinha desenvolvido uma ferramenta de análise para RdP chamado DNAmaca, contudo o PIPE se destacou pela interface mais simples de usar, plataforma independente e, uma vez que o código fonte é aberto, o aplicativo apresenta fácil extensibilidade. Desde sua criação, o aplicativo vem sendo melhorado para a correção de erros e para a adição de novas funcionalidades.

As primeiras seções dessa apostila visam a familiarização do usuário com a instalação e os recursos básicos disponíveis no PIPE, mais especificamente a versão PIPE2. Em seguida, o usuário aprende a utilizar a simulação e os módulos de análise da RdP.

2.1 Instalação inicial

O *download* do PIPE2 pode ser realizado pelo site: http://pipe2.sourceforge.net . O arquivo deve ser extraído de um arquivo 'zip'. A instalação assume que o Java já está instalado na máquina. Caso contrário, este pode ser baixado do site da *Oracle*: http://java.com (Akharware, 2005).

Independentemente de como são implementadas as ferramentas de análise da RdP, uma interface que permita um fácil desenho e edição das RdPs é fundamental para o usuário (Barnwell et al., 2004), dessa forma, nesta versão do PIPE2 as funções do *File*, *Edit*, *View*, *Draw*, *Animate* e *Help* podem ser acionadas de modo relativamente simples e intuitivo.

2.2 Funções do *File*

Dentre as funções básicas disponíveis no menu *File* tem-se:

- *New* (Crtl+N): Criar um novo documento para a RdP;
- *Open* (Crtl+O): Abrir um documento antigo;
- *Close* (Crtl+W): Fechar a guia visualizada;
- *Impor*t: Importar um documento do TimeNET;
- *Save* (Crtl+S): Salvar o documento;
- *Save as* (Crtl+Shift+S): Salvar o documento, com opção do local onde o arquivo deve ser salvo;
- *Export*: Exportar o documento com extensão PNG, PostScript ou TimeNET;
- *Print* (Crtl+P): Imprimir;
- *Exit* (Crtl+Q): Fechar o aplicativo PIPE2.

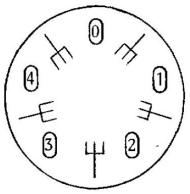


Figura 2.1: Problema Dining Philosophers de Dijkstra

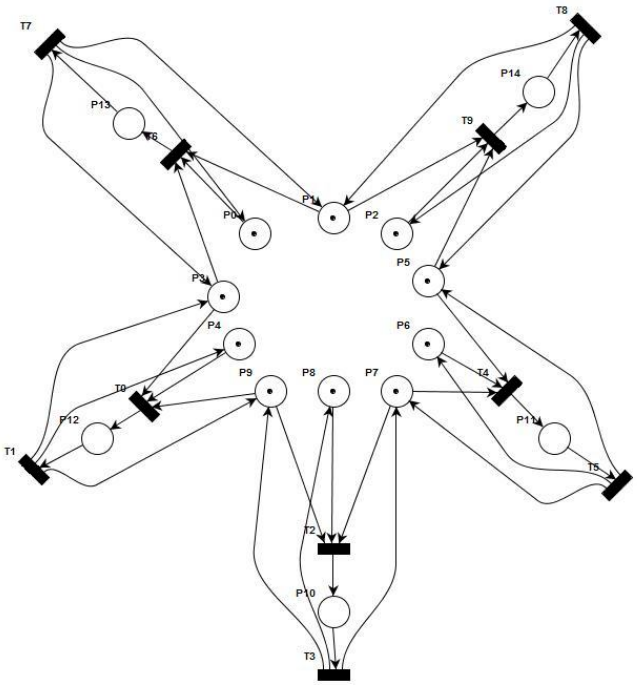


Figura 2.2: Modelo *Dining Philosophers*

2.2.1 Função *Example nets*

De forma a exemplificar o uso de RdP para representar diferentes sistemas, o aplicativo PIPE2 apresenta um banco de dados de RdPs na função *Example nets* (Barnwell et al., 2004).

Exemplo:

Dining Philosophers: Cinco filósofos moram em uma casa onde há uma mesa redonda para as refeições. Um problema, além da filosofia, é o prato servido: um tipo de espaguete difícil de manusear que para ser comido requer o uso de dois garfos (Figura 2.1). Existem dois garfos próximos a cada prato e portanto dois vizinhos não conseguirão comer simultaneamente . Como dito anteriormente, a representação gráfica dos LUGARES da RdP são circunferências; das TRANSIÇÕES, retângulos; dos ARCOS ORIENTADOS, setas; e das MARCAS, círculos dentro dos LUGARES(Dijkstra, 1971).

Na RdP do *Dining Philosophers* (Figura 2.2), os LUGARES p_1, p_3, p_5, p_7 e p_9 representam os garfos, elementos compartilhados entre os filósofos. No estado inicial, as TRANSIÇÕES t_0,t_2, t_4, t_6 e t_9 estão habilitadas. Se, por exemplo, t_2 disparar, as MARCAS em p_7, p_8 e p_9 fluirão para t_2. Em seguida, uma MARCA fluirá para o LUGAR p_{10}. Isso teria o efeito de remover 2 elementos compartilhados (garfos) da mesa, limitando assim o acesso pelos outros filósofos. Com o disparo da TRANSIÇÃO t_3, uma MARCA fluirá de p_{10} até t_3, e em seguida, MARCAS fluirão para p_7, p_8 e p_9 , retornando o sistema ao estado inicial (Chung et al., 2007). O modelo em RdP garante que apenas uma quantidade máxima de filósofos coma a cada vez .

2.3 Funções do *Edit*

Dentre as funções básicas disponíveis no menu *Edit* tem-se:

- *Undo* (Crtl+Z): Desfazer uma ação;

- *Redo* (Crtl+Y): Refazer a ação;
- *Cut* (Crtl+X): Recortar um elemento gráfico;
- *Copy* (Crtl+C): Copiar um elemento gráfico;
- *Paste* (Crtl+V): Colar um elemento gráfico;
- *Delete* (Delete): Excluir um elemento gráfico.

2.4 Funções do *View*

Dentre as funções básicas disponíveis no menu *View* tem-se:

- *Zoom out* (Crtl+Minus): Aumentar a visualização do documento;
- *Zoom in* (Crtl+Plus): Diminuir a visualização do documento;
- *Zoom*: Opção de visualizações (de 40% a 300%);
- *Cycle grid* (G): Configuração do fundo de tela quadriculado;
- *Drag* (D): Arrasta um documento para visualização (movimentação da barra de rolagem - *scrollbar*).

2.5 Funções do *Draw*

Dentre as funções básicas disponíveis no menu *Draw* tem-se:

- *Select* (S): Selecionar um elemento gráfico;
- *Place* (P): Adicionar um LUGAR;
- *Immediate transition* (I): Adicionar uma TRANSIÇÃO não temporizada - uma vez habilitada, dispara no tempo zero;
- *Timed transition* (T): Adicionar uma TRANSIÇÃO temporizada - uma vez habilitada, dispara após um tempo determinado;
- *Arc* (A): Adicionar um ARCO orientado normal;
- *Inhibitor Arc* (H): Adicionar um ARCO INIBIDOR;
- *Annotation* (N): Adicionar uma caixa de texto com comentário;
- *Add token* (NumPad+): Adicionar MARCAS de um elemento LUGAR;
- *Delete token* (NumPad-): Excluir MARCAS de um elemento LUGAR;
- *Rate Parameter* (R): Configuração de *label*;
- *Marking Parameter* (M): Configuração de *label*;

2.6 Funções do *Animate* - simulação:

Após concluir a edição da RdP, a função *Animation mode* deve ser habilitada. Desse modo, o arquivo deixará de ser editável e será habilitada a função de simulação. Para a visualização passo-a-passo do disparo das TRANSIÇÕES e do fluxo de MARCAS resultante, deve-se clicar em *Random*. Para a visualização do disparo automático de uma sequência de TRANSIÇÕES, deve-se clicar em *Animate* e definir o número desejado de disparos das TRANSIÇÕES.

Outras funções básicas:

- *Back*: Voltar ao estado anterior ao disparo da TRANSIÇÃO na simulação;
- *Forward*: Refazer o disparo da TRANSIÇÃO na simulação.

2.7 Módulos para análise das propriedades da RdP

Após concluir a edição da RdP, o aplicativo PIPE2 também permite análises qualitativas e quantitativas (Bonet et al., 2007). Essa secção apresenta os módulos de análise disponíveis:

2.7.1 *Classification*

Baseada na conexão entre os LUGARES e as TRANSIÇÕES, esse módulo permite que a RdP seja classificada nos seguintes tipos (Bonet et al., 2007):

- *State Machine* (verdadeiro ou falso) - É uma RdP ordinária (isto é, o peso dos ARCOS é sempre 1) tal que $|^\bullet t| = |t^\bullet| = 1$ para todo $t \in T$, isto é, uma RdP em que cada TRANSIÇÃO tem exatamente um LUGAR na entrada e um LUGAR na saída (Murata, 1989).
- *Marked Graph* (verdadeiro ou falso) - É uma RdP ordinária tal que $|^\bullet p| = |p^\bullet| = 1$ para todo $p \in P$, isto é, uma RdP em que cada LUGAR possui exatamente uma TRANSIÇÃO na entrada e uma TRANSIÇÃO na saída (Murata, 1989).
- *Free Choice Net* (verdadeiro ou falso) - É uma RdP ordinária tal que para todo p_1, $p_2 \in P$, $p_1{}^\bullet \cap p_2{}^\bullet \neq \varnothing \Rightarrow |p_1{}^\bullet| = |p_2{}^\bullet| \leq 1$, isto é, uma RdP em que todo ARCO saíndo de um LUGAR é o único de entrada para uma TRANSIÇÃO (Murata, 1989).
- *Extended Free Choice Net* (verdadeiro ou falso) - É uma RdP ordinária tal que $p_1{}^\bullet \cap p_2{}^\bullet \neq \varnothing \Rightarrow p_1{}^\bullet = p_2{}^\bullet$ para todo p_1, $p_2 \in P$, isto é, uma RdP em que todas as TRANSIÇÕES em conflito (isto é, que compartilham um LUGAR de entrada) possuem necessariamente os mesmos LUGARES de entrada (Murata, 1989). Ver a Figura 2.3.
- *Simple net* ou *Asymmetric choice net* (verdadeiro ou falso) - No caso de existência de TRANSIÇÕES com dois ou mais conflitos na entrada (ver a Figura 2.3): $p_1 \neq p_2 \Rightarrow p_1{}^\bullet \cap p_2{}^\bullet \neq \varnothing$ e $\{\ |p_1{}^\bullet| \leq 1$ ou $|p_2{}^\bullet| \leq 1\ \}$ para todo p_1, $p_2 \in P$, isto é, podem existir TRANSIÇÕES de saída de p_1 e p_2 em comum sendo que neste caso, existe no máximo uma TRANSIÇÃO de saída para o LUGAR p_1 ou para o LUGAR p_2 (Bause and Kritzinger, 2002).
- *Extended Simple Net* (verdadeiro ou falso) - No caso de existência de TRANSIÇÕES em conflito sendo que (ver a Figura 2.3): $p_1{}^\bullet \cap p_2{}^\bullet \neq \varnothing$ ou $p_1{}^\bullet \subseteq p_2{}^\bullet$ ou $\Rightarrow p_1{}^\bullet \supseteq p_2{}^\bullet$para todo p_1, $p_2 \in P$, isto é, uma RdP em que existem TRANSIÇÕES de saída de p_1 e p_2 em comum ou o conjunto de TRANSIÇÕES de saída de p_1 está contido ou é igual ao conjunto de TRANSIÇÕES de saída de p_2 (ou vice-versa). (Bause and Kritzinger, 2002).

2.7.2 *Comparison*

Módulo que compara as funcionalidades de duas RdPs, baseado na configuração dos LUGARES, TRANSIÇÕES, ARCOS e ARCOS INIBIDORES.

2.7.3 *Reachability Graph*

Módulo que fornece uma representação visual (grafo em forma de árvore) de todas as possibilidades (estados alcançáveis) derivados de disparos das TRANSIÇÕES de uma RdP (Bonet et al., 2007). Neste grafo (árvore de alcançabilidade), os nós representam os estados (MARCAÇÕES) da RdP e as setas entre estes nós indicam os disparos das TRANSIÇÕES da RdP (Chung et al., 2007).

2.7.4 *State Space Analysis*

Esse módulo permite a análise das propriedades qualitativas da RdP, como limitabilidade, segurança e auto-travamentos:

- Limitada (*Bounded*) - verdadeiro ou falso;
- Segura (*Safe*) - verdadeiro ou falso;
- Auto-travamento (*Deadlock*) - verdadeiro ou falso.

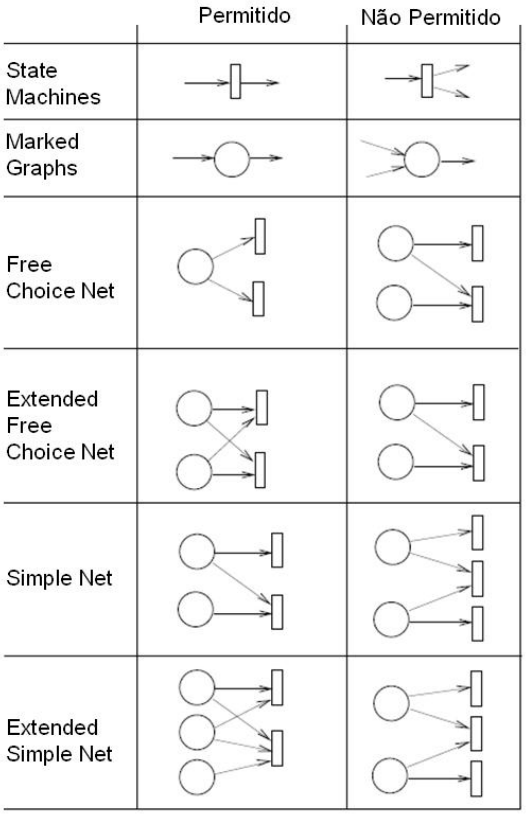


Figura 2.3: Classificação de uma Rede de Petri

2.7.5 *Minimal Siphons and Minimal traps*

Esse módulo determina o conjunto de LUGARES numa RdP em que o número de MARCAS nunca aumenta (*minimal siphons*) e o conjunto de LUGARES numa RdP que nunca fica sem suas marcas (*minimal traps*). Estes conjuntos podem ser utilizados para verificar a propriedade de vivacidade e alcançabilidade (Bonet et al., 2007).

2.7.6 *Incidence and Marking*

O comportamento dinâmico de muitos sistemas estudados na engenharia (sistemas dinâmicos com variáveis contínuas) pode ser descrito por equações diferenciais e/ou equações algébricas. Assim, é interessante descrever e analisar o comportamento dinâmico das RdPs também por equações (Murata, 1989).

Nesse contexto, este módulo disponibiliza as MATRIZES DE INCIDÊNCIA (*incidence matrices*), os vetores de MARCAÇÃO (*marking matrices*) e o vetor de disparo das TRANSIÇÕES (Bonet et al., 2007).

As matrizes de incidências são (Akharware, 2005):

- Matriz de incidencia anterior (*Backwards incidence matrix*) - Determina o número de MARCAS nos LUGARES de entrada das TRANSIÇÕES que serão consumidas quando estas TRANSIÇÕES dispararem;
- Matriz de incidencia posterior (*Forwards incidence matrix*) - Determina o número de MARCAS nos LUGARES de saída das TRANSIÇÕES que serão geradas quando estas TRANSIÇÕES dispararem.

2.7.7 *Invariant Analysis*

Esse módulo calcula os vetores dos invariantes de LUGAR (*P-invariants*) e de TRANSIÇÃO (*T-invariants*) com precisão e eficiência. Esses vetores não são difíceis de calcular uma vez que a complexidade do cálculo depende apenas do número de LUGARES e TRANSIÇÕES na RdP e não do tamanho da ÁRVORE DE ALCANÇABILIDADE (Bonet et al., 2007).

O vetor dos invariantes de LUGAR (*P-invariants*) é também chamado de *S-invariants* porque alguns autores denotam o conjunto de LUGARES por *S-set*, de "*stellen*" (LUGARES em alemão). Esse vetor é utilizado para definir se a RdP é limitada (*bounded*)(Bause and Kritzinger, 2002).

Supondo-se que a RdP está no estado M, o vetor dos invariantes de TRANSIÇÃO (*T-invariants*) define as TRANSIÇÕES que devem ser disparadas para a RdP retornar no estado M. Esse vetor é utilizado para definir se a RdP é limitada (*bounded*) e viva (*live*) (Bause and Kritzinger, 2002).

2.7.8 *GSPN Analysis*

Uma GSPN (*Generalized Stochastic Petri Net*) é uma extensão das RdPs com a adição do conceito de tempo para uma modelagem de algumas situações do mundo real (Barnwell et al., 2004). Ela possui dois tipos de TRANSIÇÕES: não temporizadas (*immediate transitions*) e temporizadas (*timed transitions*) (Akharware, 2005). O módulo de análise GSPN determina:

- Conjunto de estados alcançáveis (*Set of Tangible States*);
- Distribuição dos estados de equilíbrio dos estados alcançáveis (*Steady State Distribution of Tangible States*);
- Média do número de MARCAS em cada LUGAR (*Average Number of Tokens*);
- Densidade da probabilidade das MARCAS (*Token Probability Density*);
- Taxa de disparo das TRANSIÇÕES temporizadas (*Throughput of Timed Transitions*);
- Permanência de tempo para estados alcançáveis (*Sojourn times for tangible states*);

2.7.9 *DNAmaca*

Este módulo é uma interface com o DNAmaca, uma ferramenta para especificação, edição e solução de Cadeias de Markov, que pode ser utilizado para análise de desempenho de RdPs. O resultado produzido nesse módulo é: análise estatística de tempo das redes, que representa a distribuição do período de tempo para que o sistema evolua do estado de origem atá o estado alvo Murata (1989).

2.7.10 *Simulation*

Esse módulo calcula o número médio de MARCAS por LUGAR com intervalo de confiança de 95% para cada LUGAR da RdP. Uma abordagem analítica completa de um problema pode ser relativamente complexa e dispendiosa enquanto uma simulação devidamente projetada pode fornecer um método alternativo para identificar soluções para o problema (Bonet et al., 2007).

Assim, este módulo realiza a evolução da RdP de acordo com as regras de disparo das TRANSIÇÕES, permitindo a visualização do fluxo de MARCAS e identificação dos estados alcançáveis do sistema modelado e que é uma representação alternativa de sua dinâmica.

2.8 Exemplo

A RdP da Figura 2.4 (Bause and Kritzinger, 2002) é utilizada como exemplo. Primeiramente, obtém-se os resultados do módulo de *Classification* (Figura 2.5):

- *State Machine*: verdadeiro, pois as TRANSIÇÕES t_0, t_1, t_2 e t_3 têm exatamente e respectivamente um LUGAR na entrada (p_0, p_0, p_1 e p_2), e um LUGAR na saída (p_1, p_2, p_0 e p_0);
- *Marked Graph*: falso, pois o LUGAR p_0 possui mais de uma TRANSIÇÃO na entrada (t_2 e t_3) e mais de uma TRANSIÇÃO na saída (t_0 e t_1);
- *Free Choice Net:* verdadeiro, pois todos os ARCOS saindo dos LUGARES p_0, p_1e p_2 são os únicos de entrada para as TRANSIÇÕES t_0, t_1,t_2 e t_3(Murata, 1989);
- *Extended Free Choice Net*: verdadeiro, pois todas as TRANSIÇÕES em conflito (isto é, TRANSIÇÕES de p_0 para t_0 e p_0 para t_1) possuem necessariamente o mesmo LUGAR de entrada (p_0);
- *Simple net*: verdadeiro, pois não existem transições com conflitos na entrada que neguem essa propriedade (ver item 2.7.1 - Classification);
- *Extended Simple net:* verdadeiro, pois não existem transições com conflitos na entrada que neguem essa propriedade (ver item 2.7.1 - Classification).

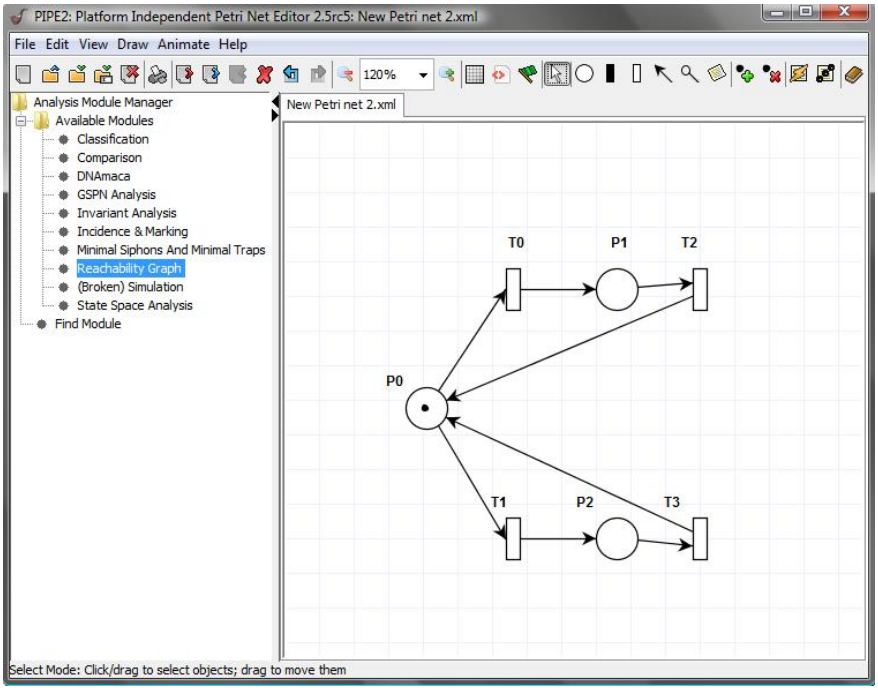


Figura 2.4: Rede de Petri - exemplo para análise.

A árvore de alcançabilidade da RdP é ilustrada na Figura 2.6. Observa-se que o módulo *Reachability Graph* gera um grafo equivalente a árvore de alcançabilidade (Figura 2.7) sendo os estados ou MARCAÇÕES $(1,0,0)$, $(0,1,0)$ e $(0,0,1)$ representados por $S0$, $S1$ e $S2$, respectivamente.

O módulo *State Space Analysis* (Figura 2.8) apresenta as propriedades qualitativas indicando que a RdP do exemplo é limitada (*bounded*), segura (*safe*) e não possui auto-travamentos (*deadlock*).

O módulo *Minimal Siphons and Minimal Traps* (Figura 2.9) determina o conjunto de LUGARES em que o número de MARCAS nunca aumenta (para o exemplo, o conjunto $\{p_0, p_1, p_2\}$) e o conjunto de LUGARES que nunca fica sem MARCAS (para o exemplo, o conjunto $\{p_0, p_1, p_2\}$).

O módulo de *Incidence and Marking*, na Figura 2.10, determina as matrizes de incidência anterior I^- e posterior I^+ . Como não existem ARCOS INIBIDORES no exemplo, a matriz H é nula. Esse módulo indica também as MARCAS no estado atual (MARCAÇÕES ou *Marking*) e as TRANSIÇÕES habilitadas (*Enabled Transitions*) para esse estado, no caso, t_0 e t_1

O módulo dos invariantes (Figura 2.11) calcula os invariantes de TRANSIÇÃO e de LUGARES. Neste exemplo, os invariantes de TRANSIÇÃO indicam que a RdP pode ser limitada e viva e os invariantes de LUGAR indicam que a RdP é limitada.

Em seguida, obtém-se os resultados do módulo GSPN *Analysis* (Figura 2.12). Este módulo indica que existem três estados alcançáveis (*Set of Tangible States*) sendo todos eles igualmente possíveis (*Steady State Distribution of Tangible States*). Considerando que não existe acréscimo ou decréscimo do número de MARCAS na RdP e que os estados alcançáveis são igualmente possíveis, então a RdP apresenta uma média de $0,\overline{3}$ MARCAS por LUGAR (*Average Number of Tokens*). O módulo de análise também indica a densidade de MARCAS por LUGAR. No caso do LUGAR p_0, nos estados M_1 e M_2 não existem MARCAS; e no estado M_0 existe uma MARCA. Dessa forma, em $0,666$ vezes dos estados, o LUGAR p_0 não apresenta MARCA ($\mu = 0$); e em$0,333$ vezes dos estados, apresenta uma MARCA ($\mu = 1$).

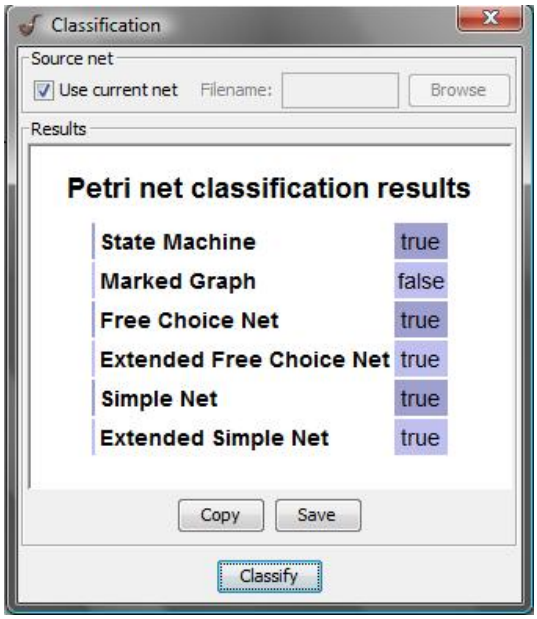


Figura 2.5: *Classification* - exemplo

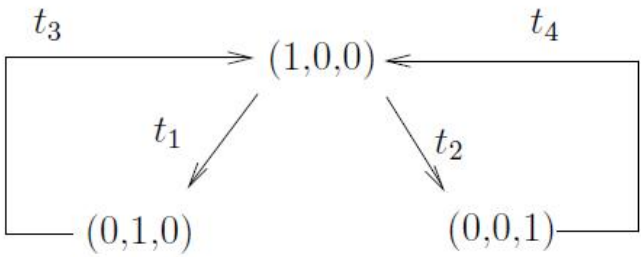


Figura 2.6: Árvore de alcançabilidade - exemplo

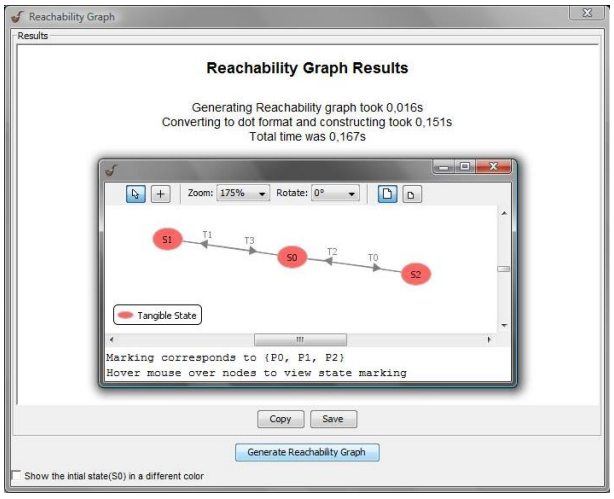


Figura 2.7: Exemplo da tela do *Reachability Graph*

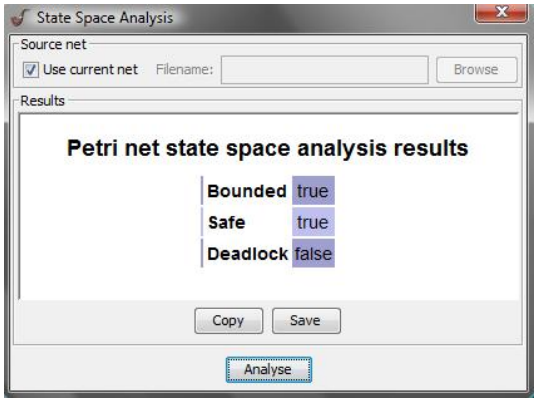


Figura 2.8: *State Space Analysis* - exemplo

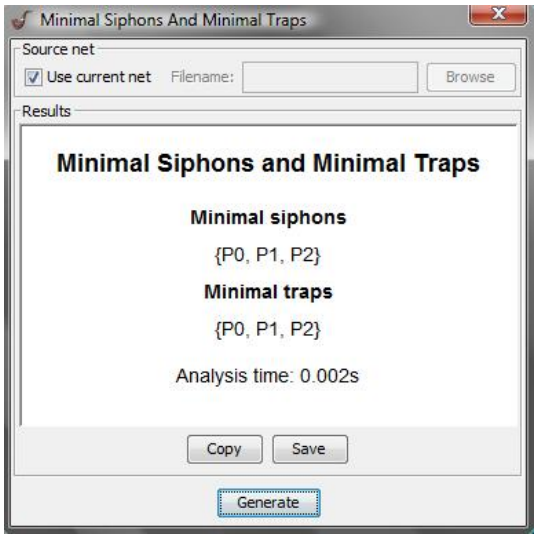


Figura 2.9: *Minimal Siphons and Minimal Traps* - exemplo

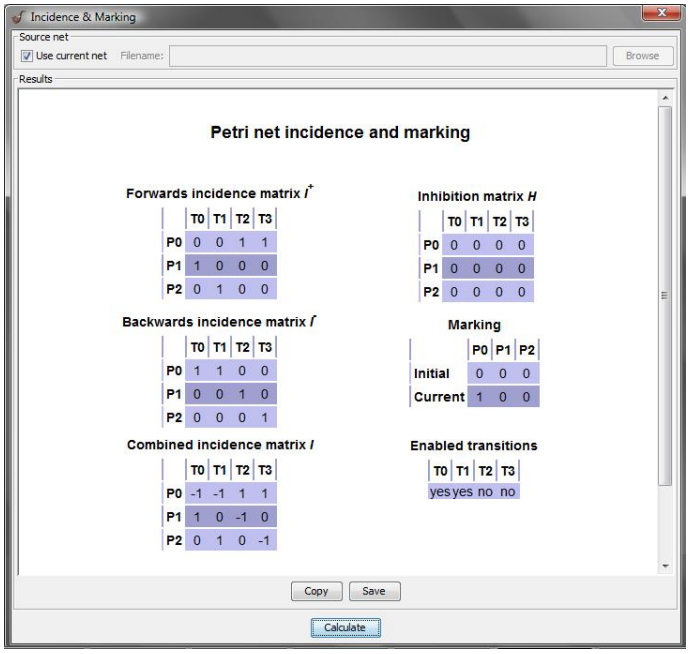


Figura 2.10: *Incidence & Marking* - exemplo

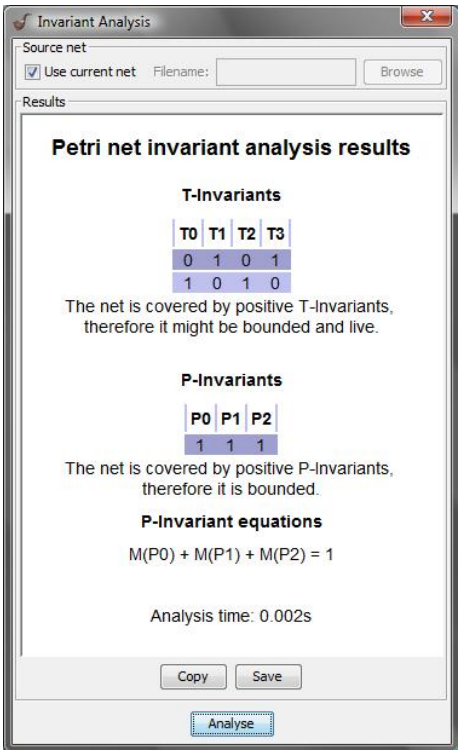


Figura 2.11: *Invariant Analysis* - exemplo

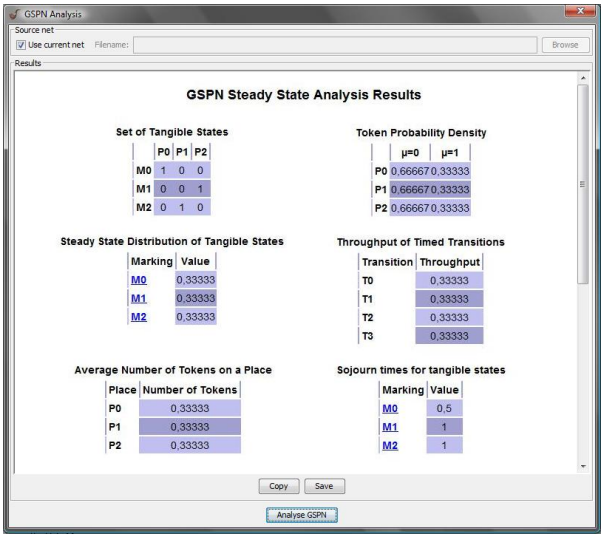


Figura 2.12: GSPN *Analysis* - exemplo

Capítulo 3

Verificação das propriedades da RdP no PIPE2

Neste capítulo é apresentado um roteiro para o aluno verificar as propriedades das RdP utilizando o aplicativo PIPE2. As propriedades a serem verificadas são: alcançabilidade, limitabilidade, segurança, vivacidade, reversibilidade, cobertura, persistência e equidade.

3.1 Alcançabilidade

Considera-se a RdP da Figura 3.1(Mourelle, 2009). Para a MARCAÇÃO inicial $(1,0,0)$, duas outras MARCAÇÕES são imediatamente alcançáveis: $(0,1,0)$ e $(1,0,1)$. Da primeira, nenhuma MARCAÇÃO é alcançável, contudo, da segunda, as MARCAÇÕES $(0,1,1)$ e $(1,0,2)$ são alcançáveis. Estas propriedades podem ser confirmadas utilizando a representação visual de todos os estados alcançáveis gerada pelo módulo *Reachability Graph*, na Figura 3.2.

3.2 Limitabilidade e Segurança

Observa-se que a RdP apresentada anteriormente (Figura 3.1) não é limitada, pois o LUGAR p_2 admite ∞ MARCAS. Essa propriedade pode ser verificada também utilizando o módulo *Reachability Graph*. Nessa representação, os detalhes de cada MARCAÇÃO são mostrados ao posicionar o cursor em cima dos nós $S0$, $S1$, $S2$ e $S3$ (Figura 3.3). Como dito anteriormente, verifica-se que da MARCAÇÃO inicial $S0$ $(1,0,0)$, duas MARCAÇÕES são alcançáveis (conectadas por setas): $S2$ $(0,1,0)$ e $S1$ $(1,0,\infty)$. Da primeira, $S2$, nenhuma MARCAÇÃO é alcançável, contudo, da segunda, as MARCAÇÕES $S3$ $(0,1,\infty)$ e $S1$ $(1,0,\infty)$ são alcançáveis. O símbolo ∞ indica que a MARCAÇÃO permite infinitas MARCAS no LUGAR. Por exemplo, $S1$ pode ser $(1,0,1)$, $(1,0,2)$, $(1,0,3)$ e assim por diante.

A visualização desse comportamento pode ser observado pelo *Animate mode*. Após selecionar essa função, visualiza-se as TRANSIÇÕES habilitadas, em cor vermelha. O disparo das TRANSIÇÕES t_0 e t_1 pode ser feito com o clique simples nelas. A partir da MARCAÇÃO inicial $(1,0,0)$, dispara-se a TRANSIÇÃO t_0. Sucessivos disparos da TRANSIÇÃO t_0 resultam no acréscimo de MARCAS no LUGAR p_2 (Figura 3.4). Dessa forma confirma-se com a simulação do PIPE2 que a RdP é não limitada.

Tem-se agora a RdP da Figura 3.5. Nessa RdP, o módulo *Reachability Graph* é também utilizado (Figura 3.6). Verifica-se que os nós $S0$, $S1$ e $S2$ representam as MARCAÇÕES $(1,0,0)$, $(0,1,0)$ e $(0,0,1)$, respectivamente. Como o número de MARCAS em cada LUGAR não excede uma MARCA, a RdP é chamada de 1-limitada e, portanto, é segura.

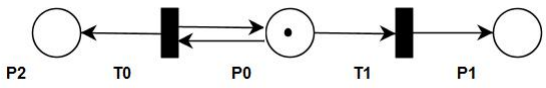


Figura 3.1: RdP para verificação da Alcançabilidade

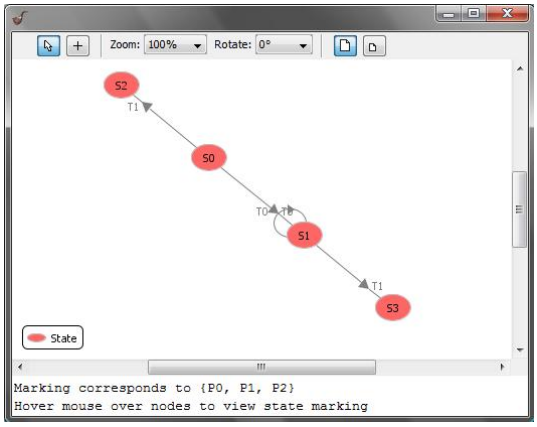


Figura 3.2: *Reachability Graph* para verificação da Alcançabilidade

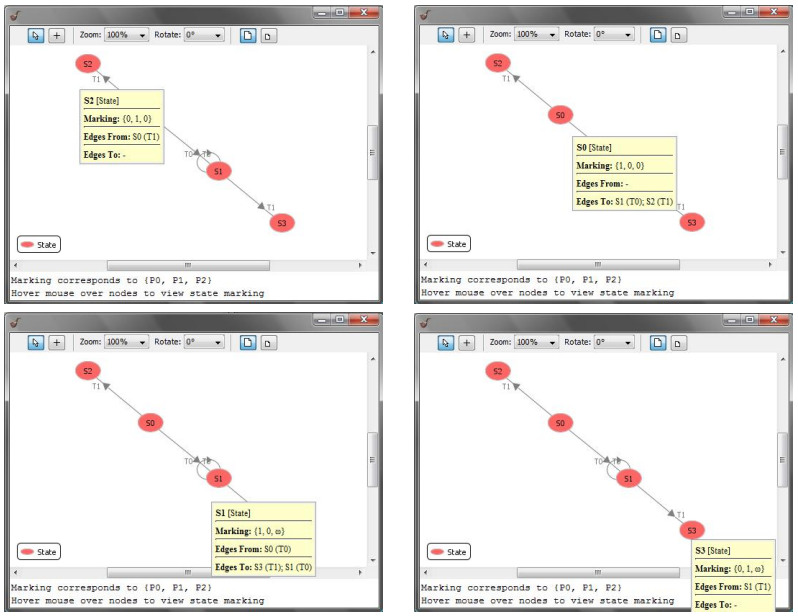


Figura 3.3: Detalhes das MARCAÇÕES possíveis na RdP

3.3 Vivacidade

Observa-se que a RdP apresentada anteriormente (Figura 3.1) é não viva, uma vez que nenhuma TRANSIÇÃO pode ser disparada se a TRANSIÇÃO t_1 disparar. A visualização desse comportamento pode ser observado pelo *Animate mode*. Após selecionar essa função, visualiza-se as TRANSIÇÕES habilitadas, em cor vermelha. O disparo das TRANSIÇÕES t_0 e t_1 pode ser feito com o clique simples nelas. A partir da MARCAÇÃO inicial $(1, 0, 0)$, dispara-se a TRANSIÇÃO t_1. Após esse disparo, as TRANSIÇÕES t_0 e t_1 não estarão mais habilitadas para o disparo e, portanto, tem-se uma condição de auto-travamento (*deadlock*) (Figura 3.7).

Comparativamente, tem-se a RdP da Figura 3.5. Essa RdP é viva, dado que não existem *deadlocks*. A visualização do comportamento da RdP pode ser observado pelo *Animate mode* (Figura 3.8). Após selecionar essa função, visualiza-se as TRANSIÇÕES habilitadas, em cor vermelha. A partir da MARCAÇÃO inicial $(1, 0, 0)$, dispara-se a TRANSIÇÃO t_0 ou a TRANSIÇÃO t_1. Após esse disparo, as TRANSIÇÕES t_0 e t_1 não estarão mais habilitadas para o disparo e a TRANSIÇÃOt_2 estará, se t_0 for disparada, ou t_3, se t_1 for disparada. Em seguida, volta-se para a MARCAÇÃO inicial $(1, 0, 0)$. Verifica-se assim que não existem situações de auto-travamento (*deadlock*) e, portanto, a RdP é viva.

Nota-se que o nível L4-viva é a condição definida para a classificação da RdP da Figura 3.5. Analisa-se a seguir o comportamento dos outros níveis de vivacidade com a RdP da Figura 3.9(Murata, 1989).

Primeiramente, analisa-se a TRANSIÇÃO t_0, classificada como L0-viva. Dado que nunca se tem ao mesmo tempo MARCA nos LUGARES p_0 e p_2 então, as pré-condições para que t_0 esteja habilitada nunca

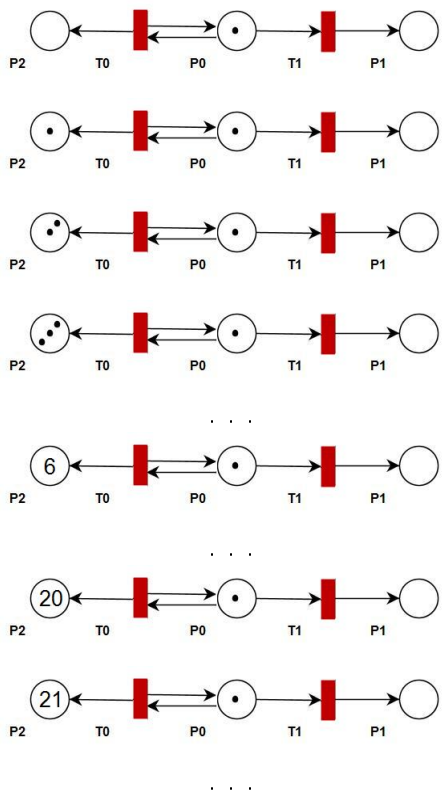


Figura 3.4: *Animate mode* para verificação da Alcançabilidade - sucessivos disparos de t_0

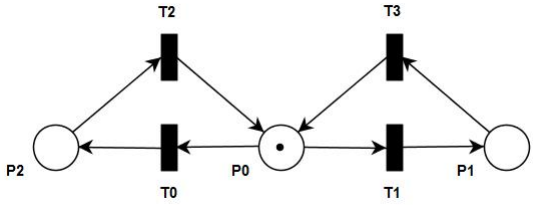


Figura 3.5: RdP para verificação da Limitabilidade

serão atendidas e, portanto, t_0 nunca poderá ser disparada. Em seguida tem-se a TRANSIÇÃO t_1, L1-viva. Pela representação visual do módulo *Reachability Graph*, verifica-se que o LUGAR p_0 nunca terá mais de uma MARCA e uma vez disparada a transição t_1 o lugar p_0 não recebe mais marcações. Sendo assim, a TRANSIÇÃO t_1 será disparada somente uma vez em qualquer sequência de disparos a partir de M_0. A TRANSIÇÃO t_2, classificada como L2-viva poderá ser disparada k vezes dependendo do número de vezes que a TRANSIÇÃO t_3 disparar (antes de t_1 ser disparada). Por fim, a TRANSIÇÃO t_3 é dita L3-viva, pois pode ser disparada infinitas vezes (antes de t_1 ser disparada).

O aluno deve verificar essas condições com a função *Animate mode*.

3.4 Reversibilidade

No exemplo da Figura 3.1, a MARCAÇÃO inicial $(1, 0, 0)$ não é alcançável após o disparo de t_0 ou t_1, portanto, a RdP não é reversível. Essa propriedade pode ser verificada pelo módulo *Reachability Graph*. Observa-se que após atingir o nó $S2$ $(0, 1, 0)$ ou $S1$ $(1, 0, \infty)$, não é possível mais retornar ao estado inicial. Comparativamente, a RdP da Figura 3.5 é reversível. Nesse caso, o módulo *Reachability Graph* indica que a MARCAÇÃO inicial $(1, 0, 0)$ é alcançável a partir das outras MARCAÇÕES.

Na mesma RdP da Figura 3.5 considera-se como estado desejado a MARCAÇÃO $(0, 1, 0)$, pode-se também classificá-la como reversível dado que é possível retornar para esta MARCAÇÃO.

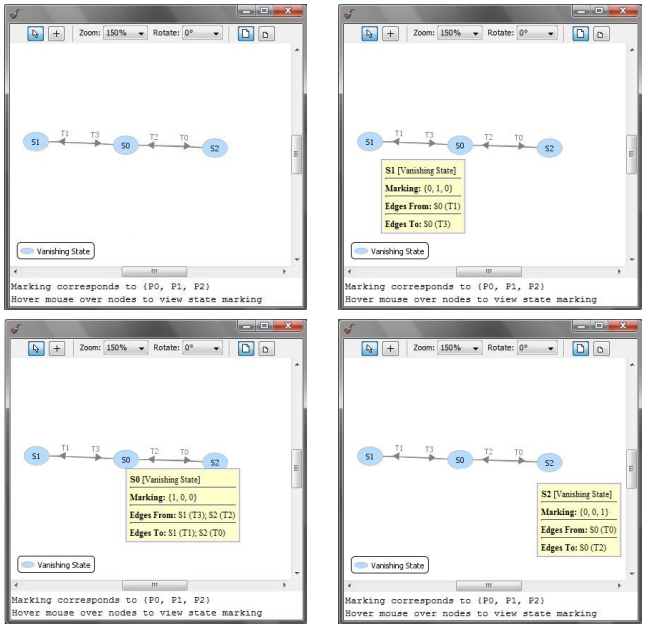


Figura 3.6: *Reachability Graph* para verificação da Limitabilidade e detalhes das MARCAÇÕES possíveis na RdP

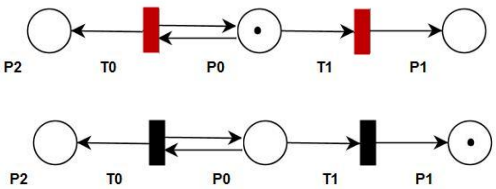


Figura 3.7: *Animate mode* para verificação da Vivacidade - disparo de $t1$ resulta em auto-travamento

3.5 Cobertura

Do exemplo da Figura 3.9, pode-se concluir que a TRANSIÇÃO t_0 é L0-viva, pois não existe uma MARCAÇÃO mínima para habilitá-la. A habilitação depende que os LUGARES p_0 e p_2 possuam MARCAS ao mesmo tempo. Comparativamente, conclui-se que t_3 é L3-viva, pois a MARCAÇÃO inicial $(1, 0, 0)$ está coberta.

3.6 Persistência

Esta propriedade pode ser observada na RdP da Figura 3.10 (Murata, 1989) com o *Animate mode*. Após selecionar essa função, o usuário visualiza a TRANSIÇÃO t_0 habilitada, em cor vermelha. O disparo da TRANSIÇÃO t_0 fará com que a MARCA flua para p_1 e p_2 e as TRANSIÇÕES t_1 e t_2 sejam habilitadas. Esta RdP é classificada como persistente, pois o disparo de t_1 não desabilita t_2 (e vice-versa) (Figura 3.11).

3.7 Equidade

Observa-se que as TRANSIÇÕES t_0 e t_1 da Figura 3.12 (Murata, 1989) são iguais-limitadas, pois o número de vezes que t_0 dispara é limitada pelo disparo de t_1. Como o par de TRANSIÇÃO desta RdP tem essa propriedade, esta RdP é classificada como igual-limitada. Além disso, dado que as TRANSIÇÕES t_0 e t_1 são habilitadas e disparadas infinitamente em uma sequencia de disparos, então a RdP também é incondicionalmente igual.

Comparativamente, a RdP da Figura 3.10 é dita incondicionalmente igual, mas não é igual-limitada, dado que não existe limite no número de vezes que a TRANSIÇÃO t_1 pode disparar sem disparar as demais enquanto a quantidade de MARCAS do LUGAR p_2 for ilimitada.

O aluno deve verificar essas propriedades com a função *Animate mode*.

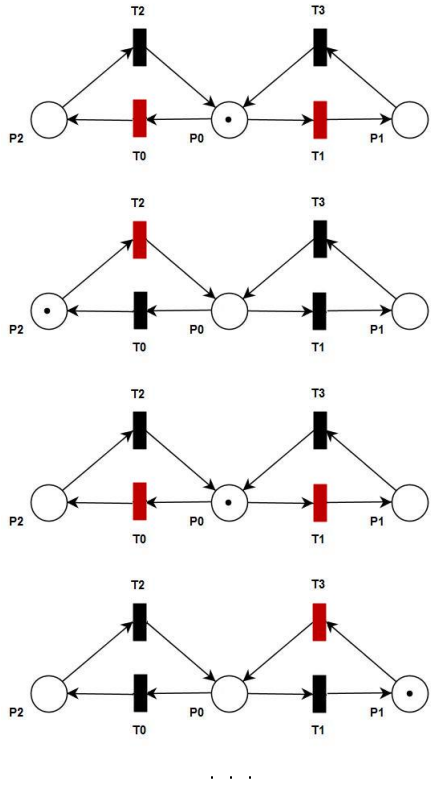


Figura 3.8: *Animate mode* para verificação da Vivacidade - ausência de *deadlocks*

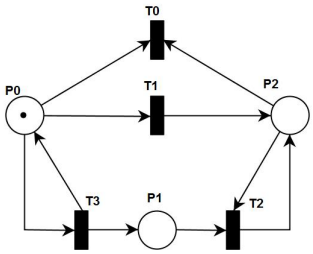


Figura 3.9: RdP para verificação da Vivacidade

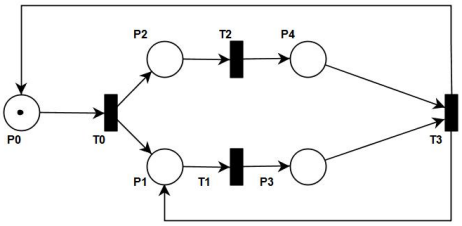


Figura 3.10: RdP para verificação da Persistência

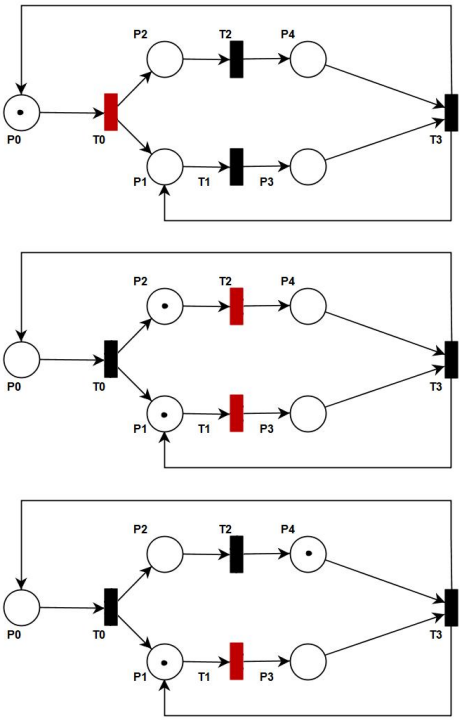


Figura 3.11: *Animate mode* para verificação da Persistência - disparo de uma transição não desabilita a outra

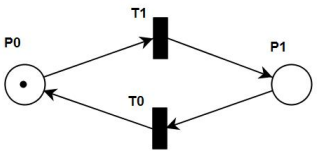
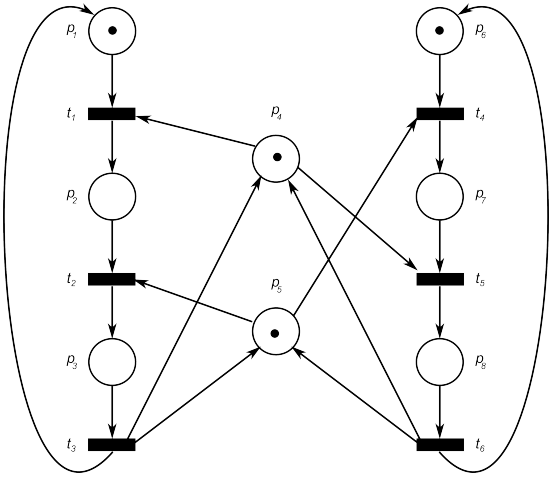


Figura 3.12: RdP para verificação da Equidade

Capítulo 4

Exercício 1: Propriedades da RdP no PIPE

Avalie o modelo representado na figura com base nas propriedades em seus devidos campos.



Alcançabilidade

Limitabilidade e segurança

Vivacidade

Reversibilidade

Cobertura

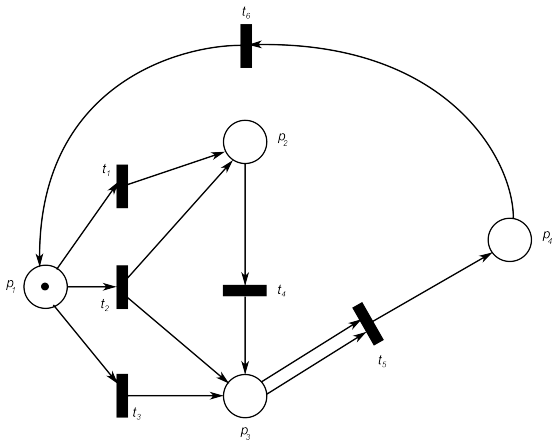
Persistência

Equidade

Capítulo 5

Exercício 2: Propriedades da RdP no PIPE

Avalie o modelo representado na figura com base nas propriedades em seus devidos campos.



Alcançabilidade

Limitabilidade e segurança

Vivacidade

Reversibilidade

Cobertura

Persistência

Equidade

Capítulo 6

Exercício 3: Comportamento dinâmico de modelos

Considera-se aqui conjuntos de dispositivos eletromecânicos e eletrônicos que emulam um sistema produtivo disperso chamado de sistema flexível de montagem automatizado (SFMA). Este SFMA é composto de 4 subsistemas (ou estações de trabalho) com funções específicas e cada um desses subsistemas pode ser entendido como um sistema produtivo (SP).

Cada subsistema do SFMA tem um conjunto de sensores e atuadores mais adequados ao seu funcionamento, isto é, execução de um processo produtivo. Estes subsistemas do SFMA são integrados tanto do ponto de vista lógico como físico. No presente caso, o processo produtivo global considerado é para a montagem de um produto que pode ser de 3 tipos:

- base prata com pino preto, mola e tampa;
- base preta com pino prata, mola e tampa; e
- base rosa com pino preto, mola e tampa.

A Figura 6.1 mostra uma foto das bases, pinos, mola e tampa.

6.1 Sistema flexível de montagem automatizada (SFMA)

Cada subsistema do SFMA deve realizar suas atividades de modo a atender a solicitação do cliente por um produto específico. Os subsistemas que compõem o SFMA são: o subsistema de alimentação, de inspeção, de montagem e de transporte. A Figura 6.2 ilustra o SFMA e seus subsistemas. Nas próximas subseções, os subsistemas são detalhados.

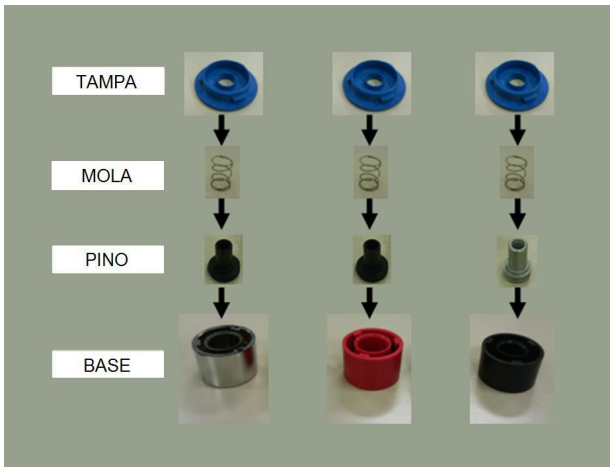


Figura 6.1: Foto da base nas três cores (prata, preta e rosa), do pino nas duas cores (preta e cinza), mola e tampa (azul).

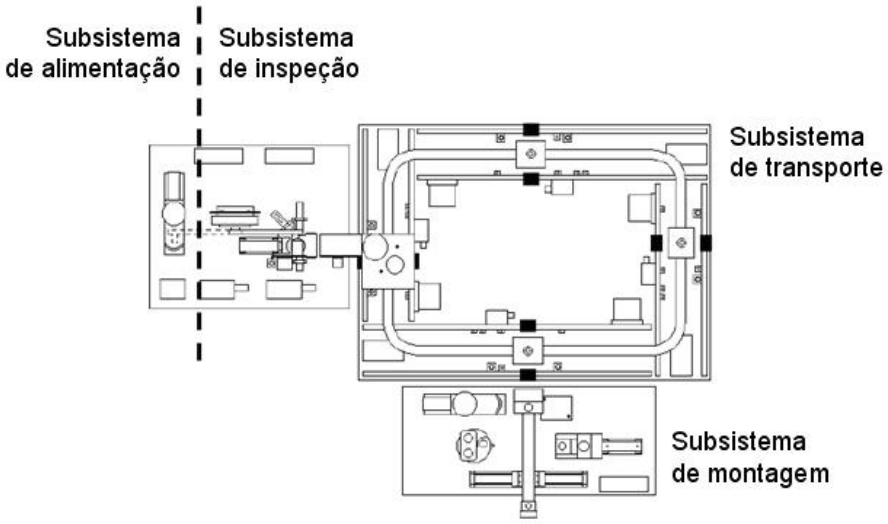


Figura 6.2: Subsistemas do SFMA.

6.2 Subsistema de alimentação

O subsistema de alimentação do SFMA emula uma instalação que provê matéria prima para outros sistemas produtivos (SPs). Essa matéria prima deve ser retirada do seu estoque e transportada para outras instalações e a função deste subsistema de alimentação é armazenar de forma otimizada e conservar as matérias primas, além da retirada dessas do estoque.

No subsistema de alimentação do SFMA, a matéria prima armazenada é a base que pode ser de vários tipos, de acordo com as cores e dimensões. O armazenamento das bases é feito em um *buffer* na forma de fila em que o subsistema de alimentação não é capaz de identificar a cor nem as dimensões das bases. Um pistão pneumático é acionado para retirar a base do *buffer* quando o subsistema de alimentação recebe a requisição de nova base. Após completar a remoção da base, um braço pneumático remove a base (prendendo-a por sucção) deste subsistema e a transporta para outro subsistema.

No caso do subsistema de alimentação, os atuadores presentes são o pistão e o braço pneumáticos. Há apenas um sensor, do tipo capacitivo, neste subsistema para detectar a presença de bases no *buffer*. Esses elementos podem ser vistos na Figura 6.3 que mostra uma representação esquemática da planta do subsistema de alimentação.

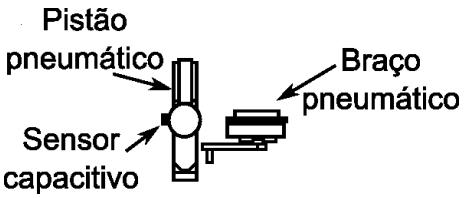


Figura 6.3: Representação esquemática do subsistema de alimentação do SFMA

6.3 Subsistema de inspeção

O subsistema de inspeção do SFMA emula uma instalação que mede e qualifica a base a ser usada em outros SPs. A base tem que atender as especificações dos clientes que solicitam o produto final.

No subsistema de inspeção do SFMA, são avaliadas a cor e a altura da base que chega. Inicialmente, a base é trazida pelo braço pneumático (ou outro meio de transporte) para a bancada de inspeção, que é o local onde estão todos os dispositivos de inspeção. Então, é feita uma inspeção de cor que utiliza para isso a informação de 3 sensores: um capacitivo, um indutivo e um óptico. O sensor capacitivo identifica a presença de uma base na bancada de inspeção. O sensor indutivo identifica se uma base na bancada de inspeção possui metais em sua composição (inclusive acabamentos metálicos). O sensor óptico identifica se uma base possui cor (diferenciando apenas as bases coloridas das peças sem cor). A

bancada de inspeção possui ainda uma plataforma que eleva a base até um sensor piezoelétrico capaz de medir a altura da peça. Após essas medições, o sistema de inspeção determina se a base atende aos requisitos do pedido. Caso estes não estejam atendidos a base é empurrada por um pistão pneumático para um *buffer* chamado de "lixo" e caso as especificações estejam atendidas a base é empurrada para uma rampa que coloca a base num *pallet* (do sistema de transporte) e que leva a outro subsistema do SFMA.

No caso do subsistema de inspeção, os atuadores presentes são o pistão pneumático e a plataforma móvel (de elevação). Quatro sensores compõem o sistema, um capacitivo, um indutivo, um óptico e um piezoelétrico. A Figura 6.4 mostra uma foto dos subsistemas de alimentação e inspeção, além de uma representação esquemática da planta do subsistema de inspeção.

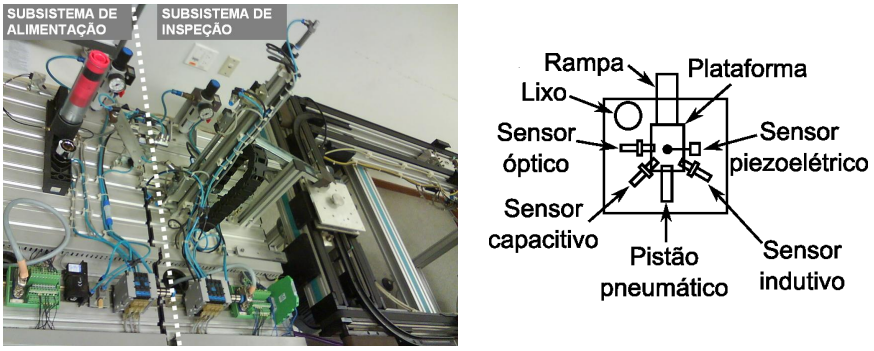


Figura 6.4: Foto dos subsistemas de alimentação e de inspeção; e a representação esquemática do subsistema de inspeção do SFMA

6.4 Subsistema de montagem

O subsistema de montagem do SFMA emula uma instalação que fabrica um produto a partir das matérias primas recebidas. Num subsistema como o de montagem existem equipamentos e manipuladores que permitem manusear, encaixar, posicionar e outras funções que alteram os estados das matérias primas inicialmente recebidas.

No subsistema de montagem do SFMA, o manipulador é um braço robótico com 3 juntas prismáticas (ou robô cartesiano) acionadas por motor elétrico e o *end-effector*, isto é a ferramenta-garra tem uma válvula pneumática para a sua abertura e fechamento. O robô cartesiano inicialmente retira a base que está num *pallet* e a coloca em uma bancada que possui um mecanismo para prendê-la e impedir qualquer movimentação da mesma. Neste subsistema tem-se um *buffer* de outra matéria prima, isto é, a mola. Neste *buffer* existe também um dispositivo acionado por um pistão pneumático para retirar uma mola do *buffer* e o posicioná-lo para ser retirada pelo robô. A mola é retirada pelo robô e levada colocada dentro da base. Neste subsistema tem-se ainda dois *buffers* de pinos, um para cada cor. A cor do pino é escolhida de acordo com a cor da base em que ele será acoplado. Um atuador pneumático de rotação remove o pino escolhido do *buffer* e posiciona-o para retirada pelo robô. O robô cartesiano então retira o pino e o coloca sobre a mola dentro da base. Por fim, um pistão pneumático remove uma tampa de um *buffer* de tampas. As tampas são iguais para qualquer peça e o robô cartesiano retira a tampa e a coloca de forma alinhada com o pino e com a mola, para que os elementos se encaixem perfeitamente. A ferramenta-garra do robô cartesiano então gira a tampa para fechar o produto final. O robô remove da bancada o produto final e posiciona-o sobre o *pallet* que vai retirá-lo do subsistema de montagem.

No subsistema de montagem, tem-se um sensor capacitivo na bancada, para detectar a presença de peça. Os atuadores são os três motores que acionam o robô cartesiano, a válvula pneumática de abertura e fechamento da ferramenta, os dois pistões pneumáticos que removem a mola e a tampa de seus *buffers* e o atuador pneumático de rotação, que retira o pino de um dos *buffers*. A Figura 6.5 mostra uma foto do subsistema de montagem, além de uma representação esquemática da planta do subsistema de montagem.

6.5 Subsistema de transporte

O subsistema de transporte do SFMA emula um SP que realiza serviços de transporte de matérias primas ou produtos. Sistemas com essas características necessitam de veículos especializados para cada tipo de

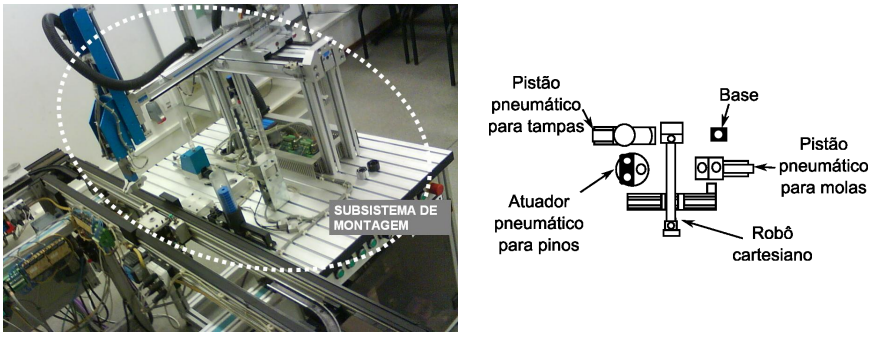


Figura 6.5: Foto do subsistema de montagem; e a representação esquemática do subsistema de montagem do SFMA

material a ser transportado. SPs que fornecem esse tipo de serviço devem se preocupar com especificações como o tempo de transporte, dimensões e características do material a ser transportado, a segurança que precisa ser considerada para evitar riscos de contaminação, explosão e outros e com a qualidade no manuseio para evitar danos ao material transportado.

No subsistema de transporte do SFMA, os veículos de transporte são *pallets* que ficam em contato com esteiras que se mantém em constante movimento. A parada do *pallet* num certo local é feita por pistões elétricos colocados nos pontos de parada do subsistema de transporte. Estes pistões impedem fisicamente o movimento dos *pallets* de acordo com sua posição: aberto ou fechado. O transporte é feito entre as estações que estão associadas aos subsistemas anteriormente citados e outros associadas a sistemas para a retirada do produto final e de armazenagem de *pallets*.

Em todo o subsistema da transporte do SFMA, há sensores capacitivos que indicam a presença dos *pallets* em cada estação. Os atuadores presentes nesse subsistema são pistões elétricos que param os *pallets* em cada estação em duas possíveis posições: na fila de entrada da estação e na área de trabalho da estação. A Figura 6.6 mostra uma foto do subsistema de transporte, além de uma representação esquemática da planta do subsistema de transporte.

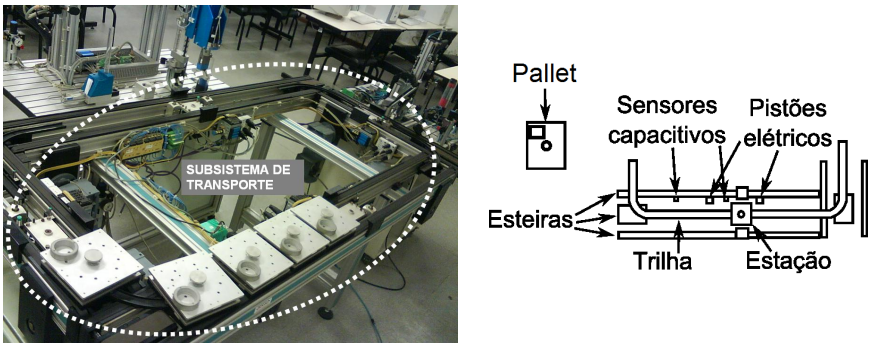


Figura 6.6: Foto do subsistema de transporte; e a representação esquemática de uma estação do subsistema de transporte do SFMA

6.6 Atividade a ser realizada

Para esse exercício, considera-se que, após o pedido feito pelo cliente, o subsistema de alimentação provê uma base para o sistema. Essa base é levada pelo braço pneumático do subsistema de alimentação para a plataforma do subsistema de inspeção. No subsistema de inspeção é identificada a cor da base, de acordo com a Tabela 6.1 de sinais dos sensores por cores da base, e só depois é medida a altura da respectiva base. Caso a altura da peça seja adequada, a base aguarda um *pallet* (do subsistema de transporte) na estação, caso contrário é jogada no "lixo". Da estação referente ao subsistema de inspeção, o *pallet* vai para a estação referente ao subsistema de montagem. Neste caso, pode haver filas de *pallets* esperando para entrar na estação e isso também deve ser modelado, ou seja, deve-se manter a ordem dos *pallets* com bases que saíram da estação referente ao subsistema de inspeção e que entram posteriormente na estação referente ao subsistema de montagem. Após a montagem de acordo com o procedimento devido e

de acordo com a Tabela 6.2, o produto final deve sair do subsistema de montagem por meio de um *pallet* que deve ir até a estação onde o produto final é encaminhado para um *buffer* de produtos finais. Este *pallet* fica então liberado para outros serviços. Os tempos em cada processo estão descritos na Tabela 6.3.

O aluno deve inicialmente desenvolver o PFS geral do processo produtivo global. Depois deve detalhar as atividades de cada um dos subsistemas ainda com o PFS. A seguir, deve-se detalhar estes PFS em rede de Petri usando o PIPE. Verificar as propriedades de cada uma das RdP geradas.

Tabela 6.1: Sinais de sensores por cor da base

Sensor indutivo	Sensor óptico	Cor da base
0	0	Preto
0	1	Rosa
1	1	Prata

Tabela 6.2: Cor do pino de acordo com as cores da base

Cor do pino	Cor da base
Prata	Preto
Preto	Rosa
	Prata

Tabela 6.3: Tempos associados às principais atividades do processo produtivo

Processo	Tempo
Retirada e posicionamento de elementos/matérias primas (bases, pinos, molas tampas) dos *buffers* nos subsistemas	1
Movimento do do braço do subsistema de alimentação para o de inspeção (transporte da base para o subsistema de inspeção)	1
Inspeção de cor no subsistema de inspeção	2
Inspeção de altura no subsistema de inspeção	1
Remoção da base para o "lixo" no subsistema de inspeção	1
Carregamento da base no subsistema de inspeção para o *pallet* do subsistema de transporte	1
Movimentação do *pallet* entre estações	5
Descarregamento (pelo robô) da base que está no *pallet* para o subsistema de montagem	1
Fixação da base na bancada do subsistema de montagem	1
Montagem (pelo robô) da mola na base que está na bancada do subsistema de montagem	3
Montagem (pelo robô) do pino na base que está na bancada do subsistema de montagem	10
Montagem (pelo robô) da tampa na base que está na bancada do subsistema de montagem	8
Carregamento do produto final (base com mola, pino e tampa devidamente montados) para o o *pallet* do subsistema de transporte	2
Descarregamento do produto final montado do *pallet* para o *buffer* de armazenagem	1

Referências Bibliográficas

N. R. Adam, V. Atluri, and W.K. Huang. Modeling and analysis of workflows using Petri nets. *Journal of Intelligent Information System*, 10(2):131–158, 1998.

N. Akharware. PIPE2: Platform independent Petri net editor. Technical report, Department of Computing - Imperial College London, 2005.

T. Barnwell, M. Camacho, M. Cook, M. Gready, P. Kyme, and M. Tsouchlaris. Final report - Petri net analyser - group 4. Technical report, Department of Computing - Imperial College London, 2004.

F. Bause and P. S. Kritzinger. *Stochastic Petri Nets - An Introduction to the Theory.* Universitšat Dortmund, 2002.

P. Bonet, C.M. Lladó, R. Puigjaner, and W.J. Knottenbelt. PIPE2.5: a Petri net tool for performance modeling. Technical report, Department of Computing - Imperial College London, 2007.

W. Brauer and W. Reisig. Carl Adam Petri and "Petri nets". *Informatik-Spektrum*, 29:369–381, 2006.

E. Chung, T. Kimber, B. Kirby, T. Master, and M. Worthington. Petri nets group project - final report. Technical report, Department of Computing - Imperial College London, 2007.

E.W. Dijkstra. Hierarchical ordering of sequential processes. *Acta Informatica*, 1:115–138, 1971.

J. I. Garcia Melo, R. A. G. Morales, and P. E. Miyagi. Supervisory system for hybrid productive systems based on Bayesian networks and OO-DPT nets. In *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, pages 1108–1111, 2008.

R. Hamadi and B. Benatallah. A Petri net-based model for web service composition. In *Proceedings of the 14th Australasian Database Conference*, pages 191–200. Australian Computer Society, 2003.

P. J. Kaneshiro, J. I. Garcia Melo, P. E. Miyagi, and C. E. Cugnasca. Modeling of collision resolution algorithm in Lonworks networks. In *Proceedings of ASME International Mechanical Engineering Congress and Exposition*, pages 743–749. ASME, 2008.

B. Kiepuszewski, A.H.M. ter Hofsted, and W.P.M. van der Aalst. Fundamentals of control flow in workflows. *Acta Informatica*, 39(3):143–209, 2003.

J.S. Lee, M.C. Zhou, and P.L. Hsu. An application of Petri nets to supervisory control for human-computer interactive systems. *IEEE Transactions on Industrial Electronics*, 52(5):1220–1226, 2005.

P. E Miyagi. *Controle Programável - Fundamentos do Controle de Sistemas a Eventos Discretos.* Editora Edgard Blücher, 1996.

R. A. G. Morales, J. I. Garcia Melo, and P. E. Miyagi. Diagnosis and treatment of faults in productive systems based on Bayesian networks and Petri net. In *IEEE International Conference on Automation Science and Engineering, CASE*, pages 357–362, 2007.

L. M. Mourelle. Controle de processos por computador - redes de Petri. Technical report, Universidade do Estado do Rio de Janeiro - Faculdade de Engenharia - Departamento de Engenharia de Sistemas e Computação, 2009.

T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of IEEE*, 77(4):541–580, 1989.

M. G. V. Nassar, J. I. Garcia Melo, P. E. Miyagi, and D. J. Santos Filho. Modeling and analysing of the material entry flow system in a pickling line process using Petri nets. In *ABCM Symposium Series in Mechatronics*, pages 444–453, 2008.

T. Yoo, B. Jeong, and H. Cho. A Petri nets based functional validation for services composition. *Expert Systems with Applications*, 37:3768–3776, 2010.

R. Zurawski and M.C. Zhou. Petri nets and industrial applications: A tutorial. *IEEE Transactions on Industrial Eletronics*, 41(6):567–583, 1994.