UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MÉTODO DE RESOLUÇÃO DE DEADLOCKS NO CONTROLE DE PRODUÇÃO DE SISTEMAS DE MANUFATURA AUTOMATIZADOS UTILIZANDO REDES DE PETRI COLORIDAS

WESLEY WILLY OLIVEIRA DE SOUZA

ORIENTADOR: PROF. DR. EDILSON REIS RODRIGUES KATO

São Carlos - SP
Julho/2011

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MÉTODO DE RESOLUÇÃO DE DEADLOCKS NO CONTROLE DE PRODUÇÃO DE SISTEMAS DE MANUFATURA AUTOMATIZADOS UTILIZANDO REDES DE PETRI COLORIDAS

WESLEY WILLY OLIVEIRA DE SOUZA

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Inteligência Artificial.
Orientador: Prof. Dr. Edilson Reis Rodrigues Kato.

São Carlos - SP
Julho/2011

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

S729mr

Souza, Wesley Willy Oliveira de.
Método de resolução de deadlocks no controle de produção de sistemas de manufatura automatizados utilizando redes de Petri coloridas / Wesley Willy Oliveira de Souza. -- São Carlos : UFSCar, 2013.
114 f.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2011.

1. Inteligência artificial. 2. Redes de Petri. 3. Automação industrial. 4. Sistemas flexíveis de manufatura. 5. Sistemas automatizados de manufatura. I. Título.

CDD: 006.3 (20ª)

Universidade Federal de São Carlos

Centro de Ciências Exatas e de Tecnologia

Programa de Pós-Graduação em Ciência da Computação

"Método de Resolução de Deadlocks no Controle de Produção de Sistemas de Manufatura Automatizados utilizando Redes de Petri Coloridas"

WESLEY WILLY OLIVEIRA DE SOUZA

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação

Membros da Banca:

Prof. Dr. Edilson Reis Rodrigues Kato
(Orientador - DC/UFSCar)

Prof. Dr. Orides Morandin Júnior
(DC/UFSCar)

Prof. Dr. Emerson Carlos Pedrino
(DC/UFSCar)

Prof. Dr. Valentin Obac Roda
(Departamento de Engenharia Elétrica/UFRN)

São Carlos
Julho/2011

Dedico às minhas sobrinhas, Manuela e Angelina e prima Raissa.

AGRADECIMENTOS

Agradeço primeiramente a Deus por me dar saúde e por colocar pessoas realmente especiais em minha vida como, por exemplo:

Meus tios José Eduardo Dinamarco Rodrigues (Tio Dú) e Teresa Carlos de Souza, pois sem o apoio moral, emocional e financeiro, a dedicação, a paciência, o carinho, os conselhos deles eu não chegaria até aqui hoje;

Minha noiva Michelle Zattoni por sempre estar ao meu lado nas horas boas e ruins, como nas vezes em que dormia nas cadeiras do laboratório para me fazer companhia enquanto eu virava as noites estudando, além de muitos outros momentos. Agradeço também pela preocupação, encorajamento e paciência (que ela sempre pedia no lugar de força). Te amo muito, linda!;

Meus sogros Wilson e Silvia pelo exemplo de vida, acolhimento e pelo carinho que nunca me faltou.

Agradeço também aos meus queridos amigos:

À Maísa Cristina Duarte (Cabeça), por aguentar minhas brincadeiras, algumas vezes chatas, me ajudar nos momentos de angústias, palhaçadas, aflições e principalmente por estar sempre por perto em todas as horas, quando precisei;

Ao Flávio Aldrovandi Montoro, por fazer questionamentos que só ele sabe fazer, para me proporcionar ser mais crítico, o que hoje sou e busco melhorar a cada dia. Sempre trabalhando juntos em [10, -23];

Ao Josué Garcia de Araújo pelas risadas, bobagens, conversas e viagens que tivemos e foram marcantes;

Aos amigos do laboratório Tear: Mayra, Marcos, Vinícius, Carlos, Felipe e ao professor Dr. Orides Morandim Jr. pelo apoio, companheirismo e orientação;

Aos amigos do mestrado (turma de 2009), pela companhia e aprendizado nas jornadas compartilhadas de estudos, principalmente nos dias em que ficamos acordados varando as noites para estudar para as provas na casa de Walter e Vanessa. Sempre lotada, muitas vezes tomando café da manhã, almoçando, lanchando e jantando e todas as demais refeições para aquele batalhão de pessoas. Valeu Walter e Vanessa por tudo!;

Ao professor e orientador Dr. Edilson Reis Rodrigues Kato pelo apoio e encorajamento na pesquisa;

Aos Doutores do departamento pelos conhecimentos transmitidos. Em especial: a Profa. Dra. Heloisa por me receber em sua sala em minha visita ao departamento;

Aos professores do curso de Ciência da Computação da Universidade Paulista (UNIP) de Ribeirão Preto: Lucas, Rodrigo, Leandro e Avelino pelo apoio e incentivo ao mestrado;

À Coordenação do Programa de Pós-graduação em Ciência da Computação pelo apoio institucional;

Finalmente e não menos importante, à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Capes) pela ajuda através da bolsa de estudos.

Deus nos fez simples e direitos, mas nós complicamos tudo

Eclesiastes 7:29

RESUMO

O Compartilhamento de recursos é uma das principais características dos Sistemas de Manufatura Automatizados (SMA), esta característica pode ser um sinônimo de vantagens, mas por outro lado pode levar ao problema de *deadlock*. Diversos trabalhos têm sugerido métodos e técnicas para reduzir ou eliminar a ocorrência *de deadlocks* em SMA. As estratégias de resolução de *deadlock* sugerido na literatura podem ser classificados nos seguintes tipos: detecção e recuperação de *deadlocks*, prevenção de *deadlocks* e de *deadlock*.

Este trabalho propõe um método de resolução para o controle de sistemas de manufatura automatizados utilizando redes de Petri coloridas, o principal objetivo do método proposto é obter um controlador da produção livre de *deadlocks*. Para modelar, simular e implementar a técnica em um controlador de produção um software dedicado é usado para modelar a rede de Petri. Inicialmente, um modelo usando Redes de Petri Coloridas é proposto, então mudanças são introduzidas, a fim de eliminar os *deadlocks*, resultando em um modelo de rede de Petri de um SMA livre de *deadlock*. Finalmente o modelo é implementado no cenário de um Sistema de Manufatura Flexível para testes e validação do método.

.

Palavras-chave: Automação Industrial, Sistemas Flexíveis de Manufatura, Sistemas de Manufatura Automatizados, Redes de Petri Coloridas.

ABSTRACT

Resources sharing is one of the main characteristics of Automated Manufacturing Systems (SMA), this characteristic can be a synonym of advantages but on the other hand it may lead to the problem of deadlock. Several papers have suggested methods and techniques to reduce or eliminate the occurrence of deadlocks in SMA systems. The deadlock resolution strategies suggested in the literature can be classified into the following types: deadlock detection and recovery, deadlock prevention and deadlock avoidance.

This work proposes a resolution method for the control of automated manufacturing systems using colored Petri nets, the main goal of the proposed method is to provide a deadlock-free production controller. To model, simulate and implement the technique in a production controller a dedicated software is used to model the Petri Nets. Initially, a model using Colored Petri Nets is proposed, then changes in the model are introduced in order to eliminate the deadlocks, resulting in a Colored Petri net control model of the SMA. Finally the model is implemented in the scenario of a Flexible Manufacturing System for testing and validation of the method.

Keywords: Industrial Automation, Flexible Manufacturing Systems, Automated Manufacturing System, Colored Petri Nets.

LISTA DE FIGURAS

Figura 1: Deadlock no tráfego de veículos - adaptado de Coffman (1971) 14

Figura 2: Mudança de estados de uma rede de Petri: (a) pelo disparo da transição t1; (b) pelo disparo da transição t2 (MAGGIO, 2005). 26

Figura 3: Uma estrutura de rede de Petri chamada de conflito. Esta estrutura apresenta um não determinismo (MURATA, 1989). 26

Figura 4: Modelo de rede de Petri 2-bounded (MURATA, 1989). 29

Figura 5: Modelo de rede de Petri representando atividades paralelas determinísticas (MURATA, 1989). 29

Figura 6: Exemplo de um modelo de rede de Petri segura e não viva, (L1-live) (MURATA, 1989). 30

Figura 7: Exemplo de construção da Árvore de Cobertura - adaptado de Oliveira Junior (2006) 32

Figura 8: Grafo de alocação de recursos. (a) Processo de posse de um recurso. (b) Processo requisitando um recurso. (c) Deadlock. - adaptado de (TANENBAUM, 1992) 34

Figura 9: Ilustração de um sifão (a) e um laço (b) (MURATA, 1999) 35

Figura 10: Exemplo de uma rede de Petri com sifões e laços (MURATA, 1999) 36

Figura 11: Teste de redundância proposto por Uzam (2007). 39

Figura 12*: Controle de estados adicionando em uma sink transition (HUANG, 2007)* 40

Figura 13: SMA de montagem com robôs proposto por Zhou e DiCesare(1993) 42

Figura 14: Modelo de rede de Petri do SMA usado por Fanti (2004) 44

Figura 15: Cenário abordado por Wu (2008) 50

Figura 16: Processo de montagem de dois produtos concorrentes, proposto por Wu (2008) 50

Figura 17: Modelo de rede de Petri para (a) Recurso -H e (b) Recurso G 51

Figura 18: Modelo de rede de Petri para movimentação de peças por robô proposto por Wu (2008) 51

Figura 19: redes de Petri primárias por estação de trabalho. Propostas por Wu (2008) 52

Figura 20: Sub rede para montagem do produto-A 53

Figura 21: Sub rede para montagem do produto-B 53

Figura 22: Modelo de rede de Petri orientada a recursos resultante 54

Figura 23: Subrede para operações de montagem por Wu (2008) 56

Figura 24: Célula de Manufatura Flexível usado por Ezpeleta (2002) 57

Figura 25: Relação de permissividade entre as políticas de controle *Deadlock Avoidance* em SMA 58
Figura 26: Estrutura do controle 62
Figura 27: Representação gráfica de uma rede de Petri pelo CPNTools 63
Figura 28: Operação de montagem 64
Figura 29: Processo de transporte 64
Figura 30: Exemplo para roteiro de produção 65
Figura 31: Conflito entre etapas de produção 65
Figura 32: Etapas de produção sem conflitos 66
Figura 33: Cenário com três estações de produção e um robô de movimentação 66
Figura 34: Roteiros de produção do cenário de exemplo 67
Figura 35; Roteiro de produção do produto p_1 67
Figura 36: Roteiro de produção do produto p_2 68
Figura 37: Fusão dos roteiros de produção dos produtos p_1 e p_2 68
Figura 38: Rotas de produção dos produtos p1 e p2 sem conflito 69
Figura 39: Buffers sem limitações de armazenamento 70
Figura 40: Limitação de buffers 71
Figura 41: Rotas de produção dos produtos p1 e p2 sem conflito: Exemplo de módulo em uma rede de Petri 72
Figura 42: Módulo de entrada e saída do controlador 72
Figura 43: Cenário de exemplo para a técnica de DA proposta 73
Figura 44: Rotas dos produtos p_1 e p_2 73
Figura 45: Modelo de CTPN resultante do cenário de exemplo 74
Figura 46: Estado de *deadlock* da CPN na marcação *M6* 76
Figura 47: Transições que geram marcações na região crítica 77
Figura 48: Modelo de CTPN do exemplo após a aplicação da técnica de DA proposta 78
Figura 49: Arranjo do cenário proposto por Piroddi, Cordone e Fumagalli (2008). 81
Figura 50: Roteiros de produção dos produtos P1, P2 e P3. 81
Figura 51 : Modelo de rede de Petri Colorida gerado. 82
Figura 52: Ilustração do comportamento do FMS dada a ordem de produção da Tabela 5. . 83
Figura 53: Elementos relacionados à estação de trabalho w_1 85
Figura 54: Elementos relacionados à estação de trabalho w_2 86
Figura 55: Elementos relacionados à estação de trabalho w_3 86
Figura 56: Elementos relacionados ao robô r_1 87
Figura 57: Elementos relacionados ao robô r_2 88

Figura 58: Região Crítica 1 89
Figura 59: Região Crítica 2 89
Figura 60: Conjuntos de cores, variáveis, constantes e funções 90
Figura 61: Modelo de CTPN resultante 91
Figura 62 : Janela de apresentação do instalador do CPN Tools 3.2.2. 108
Figura 63: Janela de configuração do diretório para a instalação. 108
Figura 64: Tela de seleção de componentes opcionais. 109
Figura 65: Janela inicial do CPN Tools 110

LISTA DE TABELAS

Tabela 1: Elementos da rede de Petri (MURATA, 1989) 25
Tabela 2: Evolução das marcações de M0 à MF 75
Tabela 3: Evolução das marcações do modelo 75
Tabela 4: Evolução das marcações com a técnica aplicada 77
Tabela 5: Relação Transição/Evento do SMA 94
Tabela 6: Saída da simulação 1 94
Tabela 7: Saída da simulação 2 95
Tabela 8: : Saída da simulação 3 95
Tabela 9: Saída da simulação 6 96
Tabela 10: Saída da simulação 5 97
Tabela 11: Saída da simulação 6 98

LISTA DE ABREVIATURAS E SIGLAS

SMA – *Sistemas de Manufatura Automatizados*

CPN - *Colored Petri Net*

CTPN – *Colored Timed Petri Net*

DA – *Deadlock Avoidance*

DDR – *Deadlock Detection and Recovery*

DP – *Deadlock Prevention*

FMS – *Flexible Manufacturing System*

NP – *No Polynomial*

PN – *Petri Net*

S^2P – *Simple Sequential Process*

TPN – *Timed Petri Net*

VPN – *Virtual Petri Net*

RAS – *Resource Allocation System*

ROPN - *Resource-Oriented Petri Net*

FAS - *Flexible Assembled System*

SUMÁRIO

CAPÍTULO 1 - INTRODUÇÃO 14

1.1 Contextualização 14
1.2 Objetivo 16
1.3 Justificativa e Motivação 17
1.4 Estrutura do Trabalho 18

CONCEITOS FUNDAMENTAIS 20

2.1 Considerações Iniciais 20
2.2 Definições de Sistemas de Manufatura Automatizados 20
2.2.1 Elementos de um SMA 21
2.2.2 Planejamento e controle 22
2.2.3 Sistemas Flexíveis de Manufatura 23
2.2.4 Métodos de modelagem 24
2.3 Definições de redes de Petri 24
2.3.1 Extensões de redes de Petri 26
2.3.2 Rede de Petri Colorida 27
2.3.3 Rede de Petri Temporizada 27
2.3.4 Propriedades Comportamentais em redes de Petri 28
Reachability (Alcance) 28
Boundeeness (Limitação) 29
Liveness (Vivacidade) 29
Reversibility (Reversibilidade) 31
2.3.5 Métodos de análises de redes de Petri 31
2.3.6 Softwares de modelagem de redes de Petri 32

TRABALHOS RELACIONADOS 33

3.1 Considerações Iniciais 33
3.2 Detecção e Recuperação de *deadlock* 37
3.3 Deadlock Prevention 38
3.4 Deadlock Avoidance 41
3.4.1 Resolução de *deadlock* com serviço de recursos conjuntivos 41
Método de modelagem. 42
Descrição da técnica 44

Resultados 48
3.4.2 Resolução de *deadlock* usando redes de Petri orientada a recursos 49
Método de modelagem. 49
Descrição da técnica. 54
Resultados 56
3.4.3 Outras Abordagens 56
Teoria dos grafos 56
Algoritmo do banqueiro 57
3.5 Considerações Finais 58

PROPOSTA 60

4.1 Considerações Iniciais 60
4.2 Restrições abordadas 60
4.3 Estrutura do controle do sistema 61
4.4 Método de modelagem 62
4.4.1 Ordem de produção 63
4.4.2 Operação 63
4.4.3 Movimentação e Transporte 64
4.4.4 Etapas de produção 64
4.4.5 Fusão de roteiros de produção 66
4.4.6 Limitações 69
4.4.7 Módulos 71
4.5 Técnica de Resolução de *Deadlock* 72
4.6 Considerações finais 78

IMPLEMENTAÇÃO 80

5.1 Considerações Iniciais 80
5.1.1 Aplicação e análise comportamental dos resultados 80
5.2 Aplicação e análise formal dos resultados 84
5.3 Teste de validação 92
Simulação 1 94
Simulação 2 95
Simulação 3 95
Simulação 4 96
Simulação 5 97
Simulação 6 98
5.4 Considerações finais 99

CONCLUSÕES 100

6.1 Trabalhos Futuros 101

REFERÊNCIAS 102

APÊNDICE A 107

6.2 Considerações Iniciais 107
6.3 Download e instalação 107
6.4 Interface gráfica 109
6.4.1 Index 109
6.4.2 Menus de Contexto 110
6.4.3 Paletas 111
Criação 111
Estilos 112
Visualização 112
Simulação 112
6.5 Criação e edição 113
6.6 Considerações finais 114

Capítulo 1

INTRODUÇÃO

1.1 Contextualização

A busca pela competitividade no meio industrial tornou-se umas das características mais marcantes na evolução de sistemas produtivos. Com isso, o sucesso competitivo se tornou decorrente de determinados objetivos de desempenho, como qualidade, custo, flexibilidade, rapidez e confiabilidade (AGUIRRE, 2007).

Uma das principais características dos Sistemas de Manufatura Automatizados (SMA) é o compartilhamento de recursos, assim como, máquinas ou estações de produção, robôs e veículos para o transporte dos produtos, peças ou matérias-primas. Esse compartilhamento traz vários benefícios para a empresa como, por exemplo, o aumento do desempenho, flexibilidade, entre outros. Por outro lado, o compartilhamento de recursos pode acarretar no problema de *deadlock* (FANTI e ZHOU, 2002).

Deadlock é uma situação na qual um conjunto de tarefas ou processos esta esperando por um evento que somente outro processo desse mesmo conjunto pode fornecer. (TANENBAUM, 1992). A Figura 1 apresenta um exemplo de *deadlock* no tráfego.

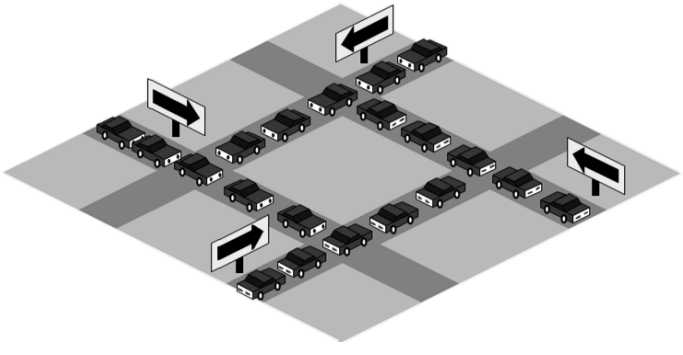


Figura 1: Deadlock no tráfego de veículos - adaptado de Coffman (1971)

Nesse caso, um carro está ocupando um recurso, que é um trecho da pista em um cruzamento, impedindo que outro carro use o mesmo trecho, mas o primeiro carro não vai liberar o recurso, pois este precisa ocupar outro trecho que está ocupado por um terceiro carro, e assim segue até que a cadeia volte para o primeiro veículo.

Deadlock é um fenômeno que acontece em sistemas onde há compartilhamento de recursos, assim como:

- Sistemas Operacionais;
- Cruzamentos de trânsito;
- Linhas férreas;
- SMA;
- Entre outros.

O foco desta proposta é abordar o problema de *deadlock* no controle de produção de SMA.

Coffman (1971) definiu quatro tipos de condições de *deadlock* que serão detalhadas no Capítulo 2, as condições são:

- Condição de exclusão mútua;
- Condição de posse e espera;
- Condição de não preempção;
- Condição de espera circular.

Inicialmente, o tratamento de *deadlock* no controle de produção de SMA era simplesmente ignorado. Quando ocorria um *deadlock,* os operadores paravam todo o sistema, abortavam as tarefas, descartavam os produtos não concluídos e reiniciavam o processo de produção. Com o tempo, as estruturas dos SMA passaram a ficar mais complexas e com isso a ocorrência de *deadlocks* aumentou para essas novas estruturas. Nos últimos anos, novas técnicas para solucionar o problema surgiram para reduzir as perdas e o tempo gasto com a tentativa de recuperar o sistema.

Atualmente na literatura são encontradas três estratégias de resolução para o problema de *deadlock* em SMA, que serão apresentadas a seguir.

Deadlock Detection and Recovery (DR) foi uma das primeiras estratégias adotadas para solucionar o problema. Nessa estratégia é necessário identificar estados do sistema de manufatura onde ocorreu um *deadlock*, em seguida, algumas decisões predeterminadas são feitas para tirar o bloqueio do processo, assim como, anular algumas tarefas que estão fortemente relacionados ao bloqueio. Para que essa estratégia seja aplicada é necessário ter informações de todos os estados do sistema em tempo de operação para que o *deadlock* seja detectado. Para solucionar o problema, primeiro ele tem que ocorrer, ou seja, o processo de produção vai parar, então as decisões tomadas levarão um tempo até que

sejam executadas e concluídas e só então o processo de produção pode voltar a ser executado, isso, em alguns casos, leva muito tempo, podendo trazer prejuízo para a empresa, além de perda de matéria-prima que poderão ser descartadas.

Deadlock Prevention (DP) é uma estratégia de resolução de *deadlock* que trata o problema antes da produção iniciar, em outras palavras, a estratégia trata o problema *off-line* antes da produção entrar em execução, ou seja, na fase de programação da produção. Para aplicar a estratégia de DP é necessário ter informações de também todos os estados e condições possíveis do sistema. A partir dessas informações, regras ou políticas podem ser aplicadas na modelagem garantindo um sistema livre de *deadlock* no controle da produção.

Outra estratégia utilizada é a *Deadlock Avoidance* (DA), essa estratégia, tem como objetivo, principal, anular eventos que dão condições para que ocorra *deadlock* dinamicamente, ou seja, em tempo de operação. A estratégia é aplicada no sistema de controle da produção, para isso é necessário que o controle tenha informações de todos os estados do sistema, e todas as tarefas de produção que estão ou ainda estarão ocorrendo em tempo de operação para que decisões de permissão ou inibição de eventos sejam tomadas (ELMEKKAWY e ELMARAGHY, 2003).

Algumas das técnicas de modelagem e análise do controle dos SMA são baseadas na teoria estrutural de redes de Petri. Em particular, as redes de Petri são adequadas para descrever a sequência de eventos que seguem a dinâmica do SMA, bem como a relação de precedência que caracterizam a ordem dos processos ou tarefas utilizando o processamento do recurso (FANTI e ZHOU, 2002).

Baseado nos conceitos estudados sobre ocorrências de *deadlocks* em SMA e nas formas de resolução existentes, este trabalho propõe um método de resolução de *deadlock* em SMA utilizando uma ferramenta de modelagem de redes de Petri coloridas chamada CPN Tools. O método resume-se em regras de modelagem resultando um modelo de rede de Petri colorida livre de *deadlocks* para o controle de SMA. O modelo resultante pode fazer parte do controle de um determinado SMA sem que este entre em um estado de *deadlock* dentro das condições e do escopo em que o sistema e o controlador foram planejados para controlar.

1.2 Objetivo

O objetivo principal desse trabalho é propor um método de resolução de *deadlocks* no controle de sistemas de manufatura automatizados utilizando as redes de Petri Coloridas.

Esse método aborda as estratégias de *deadlock* existentes para o projeto e implementação do controle da produção, em Sistemas de Manufatura Automatizados.

Outros objetivos são:

- Obter a modelagem do controle de Sistemas de Manufatura Automatizados em Redes de Petri Coloridas;
- Estabelecer um roteiro de testes e validação desses sistemas na fase de projeto;
- Obter um controlador baseado em rede de Petri Colorida livre de *deadlocks*.

1.3 Justificativa e Motivação

Com o passar dos anos e com o avanço da tecnologia, as estruturas dos SMA passaram a ficar mais complexas e com isso as ocorrências de *deadlock,* as quais normalmente nem eram tratadas*,* passaram a ficar mais frequentes, gerando prejuízos para as empresas manufatureiras. Isso trouxe consequências nas estratégias de resolução de *deadlocks*, como, por exemplo, o uso da estratégia DR para corrigir os *deadlocks* a medida que ocorrem no sistema, a estratégia DP, a qual estuda e tenta resolver os *deadlocks* já no projeto e a estratégia DA, que em operação tenta evitar os *deadlocks* de sua possível ocorrência. Em muitos casos a estratégia DR passou a ser menos eficaz em relação às outras devido as exigências de desempenho do sistema como um todo e as estratégias de *Deadlock Avoidance* e *Prevention* ficaram mais complexas de serem implementadas, uma vez que identificar todas as condições de *deadlock* e sua forma de representação se tornou uma tarefa difícil.

Vários trabalhos têm tentado sugerir métodos e técnicas que possam diminuir ou eliminar a ocorrência de *deadlocks* no sistema, buscando o melhor desempenho possível do sistema automático.

Em muitas dessas propostas algumas ferramentas tem se destacado na modelagem do sistema de manufatura e seu controle. A rede de Petri é uma delas (FANTI, MAIONE e TURCHIANO, 2000). Redes de Petri estão sendo muito utilizadas para modelagem, análises, simulações e controle de SMA. Elas se destacam pelas seguintes características (MURATA, 1989), (DESROCHERS e AL-JAAR, 1995):

- Redes de Petri capturam relações de precedência e interações estruturais de eventos estocásticos, concorrentes, e assíncronos. Além disso, a sua forma gráfica ajuda a visualizar sistemas complexos;

- Conflitos e tamanhos de *buffer* podem ser modelados de forma fácil e eficiente;
- *Deadlocks* podem ser detectados no sistema;
- Modelos de redes de Petri representam uma ferramenta de modelagem hierárquica com uma base bem desenvolvida com fundamentos matemático e prático;
- Diversas extensões de redes de Petri, assim como redes de Petri Temporizadas, redes de Petri Estocásticas (temporizadas), redes de Petri Coloridas e redes predicado / transição, permitem análises qualitativas e quantitativas da utilização de recursos, efeitos das falhas, e a taxa de transferência, entre outros;
- Modelos de rede de Petri retornam um quadro estruturado para a realização de uma análise sistemática de sistemas complexos. Vários pacotes de software foram desenvolvidos para esta finalidade;
- Finalmente, os modelos de rede de Petri também podem ser usados para implementar sistemas de controle em tempo real.

Dentre as várias extensões das redes de Petri, as redes de Petri Coloridas destacam-se na modelagem do controle de sistemas de manufatura devido às propriedades de se abstrair diferentes elementos na forma de *tokens*, tais como peças diferentes transitando no sistema. O uso de redes de Petri Coloridas na modelagem de SMA pode auxiliar na prevenção de *deadlocks* em Sistemas de manufatura complexos, podem tornar fácil a tarefa de identificação das condições de *deadlock* e a implementação de uma estratégia de resolução (FANTI, MAIONE e TURCHIANO, 2000).

O Laboratório de Pesquisa e Inovação em Tecnologias e Estratégias de Automação (Tear) do Departamento de Computação da Universidade Federal de São Carlos (UFSCar) desenvolve pesquisas em modelagem, planejamento, controle de sistemas de manufatura, entre outras. A estratégia de implementação de sistemas de controle de SMA para sistemas produtivos complexos é um dos temas abordados pelo laboratório. Estes fatores e os citados acima promoveram a motivação para que dentro dessa linha de pesquisa, fosse desenvolvido um trabalho de especificação de uma estratégia no controle de sistemas de manufatura automáticos.

1.4 Estrutura do Trabalho

No Capítulo 2 são apresentados alguns conceitos sobre Sistemas de Manufatura Automatizados, assim como suas características, os elementos que os compõem, as formas

de modelagem e outros. Em seguida serão apresentados alguns conceitos sobre redes de Petri, assim como características, elementos que a compõe, propriedades comportamentais, métodos de análises, algumas extensões e ferramentas de modelagem de redes de Petri. Finalizando o capítulo, são apresentadas as definições de *deadlock*, assim como, as condições para a ocorrência de deadlock e as estratégias de resolução mais utilizadas.

No Capítulo 3 é abordado o problema de *deadlock* em Sistemas de Manufatura Automatizados, as estratégias encontradas na literatura nos últimos anos, além disso, também são apresentados alguns dos métodos de modelagem e implementação das técnicas.

No Capítulo 4 a proposta é apresentada, assim como, a estrutura do controle do sistema onde a estratégia é implementada, o método de modelagem e a descrição da técnica de resolução de *deadlocks* proposta.

No Capítulo 5 a estratégia de resolução proposta é implementada em, os resultados obtidos são analisados e validados.

No Capítulo 6 são abordadas as contribuições deste trabalho, assim como as conclusões da pesquisa, da proposta, discussões sobre o trabalho e recomendações para trabalhos futuros.

Capítulo 2

CONCEITOS FUNDAMENTAIS

2.1 Considerações Iniciais

Este é um capítulo de conceituação aborda as linhas de pesquisa fundamentais para o entendimento do trabalho. O capítulo é dividido em três partes, na primeira parte são tratadas as definições de sistemas de manufatura, assim como os elementos que os compõem, as formas de modelagem e outros. Na segunda parte do capítulo são abordados os principais conceitos sobre redes de Petri, assim como suas extensões, variações, propriedades e aplicações. Finalizando o capítulo, na terceira parte são tratadas as definições de *deadlock*, onde podem ocorrer e as estratégias de resolução mais utilizadas.

2.2 Definições de Sistemas de Manufatura Automatizados

Os Sistemas de Manufatura Automatizados (SMA) vem sendo apresentados como opção de tecnologia na automação dos processos condizente com as características de produção de significativa parcela do mercado. Neste tipo de sistema, a versatilidade é obtida em diferentes aspectos, possibilitando uma série de benefícios para as empresas.

SMA são sistemas onde há automatização em etapas do processo de produção, ou seja, nas estações de processamentos, no sistema de transporte e no sistema de controle.

2.2.1 Elementos de um SMA

Um SMA possui componentes típicos que podem ser classificados em três categorias principais (MONTEVECHI et al., 2007) (GROOVER, 2007):

- **Estações de processamento;**

  Geralmente são máquinas que realizam operações de usinagens de peças. O tipo de máquinas pode definir se o SMA é um sistema dedicado ou aleatório.

  Sistemas dedicados desenvolvem operações específicas, ou seja, uma variedade limitada de processamentos específicos pré-programados.

  Sistemas Aleatórios podem se adaptar a uma grande variedade de peças em sequências que não são pré-programadas, ou seja, as máquinas-ferramentas devem possuir certa flexibilidade permitindo que novos projetos de peças possam ser futuramente introduzidos no sistema.

- **Sistema de movimentação e armazenagem de materiais;**

  Essa categoria representa vários tipos de equipamentos de manipulação de material automáticos, que são utilizados para transportar peças entre as estações de processamento, locais de armazenagem e pontos de entrega, são agrupados em: robôs industriais, veículos autoguiados, transportadores e armazéns automatizados. Esses sistemas devem possuir movimento aleatório entre as estações de processamento, ou seja, devem ser capazes de fluir de uma estação de trabalho para outra qualquer caso seja necessário movimentar peças paletizadas, que são montadas em *pallets* (ou paletes) de fixação. Além dessa característica apresentada, os sistemas de movimentação e armazenagem de materiais devem possuir armazenamento temporário, compatibilidade com o controle computadorizado, devem ser expansíveis em base modular, devem prover acesso distribuído no nível do chão para as estações de processamento individuais na linha de produção.

- **Sistema de controle por computador.**

  Esse sistema é usado para coordenar as atividades das estações de processamento e o sistema de movimentação e armazenagem de materiais. Segundo Montevechi et al. (2007) as funções realizadas pelo sistema de controle por computador podem ser agrupadas como segue:

  - **Armazenamento e distribuição de programa de usinagem**;

Refere-se aos programas que são armazenados e distribuídos para as diversas estações de processamento.

- **Controle de produção**;

  Essa função inclui decisões sobre troca de partes e frequência de entrada das várias peças no sistema. Atende definições de outras áreas da empresa, como, por exemplo, executar a fabricação de peças seguindo a programação de produção definida.

- **Controle de tráfego**;

  É a função que coordena o sistema primário de transporte de peças entre as estações de processamento.

- **Controle shuttle**;

  Coordena a parte secundária do sistema de movimentação para cada máquina-ferramenta, ou seja, está ligada ao sistema de carga e descarga de cada máquina do sistema.

- **Monitoramento do sistema de movimentação**;

  O computador monitora o estado de cada *pallet* nos sistemas primário e secundário de movimentação e também o estado de cada peça existente no sistema.

- **Controle de ferramenta**;

  Essa função possui dois aspectos: localização de ferramenta e monitoração de tempo de vida útil.

- **Monitoramento e informação de desempenho do sistema**.

  Função que avalia o desempenho do sistema com relação a parâmetros predeterminados que possam ser utilizados como realimentação para o controle.

2.2.2 Planejamento e controle

Para aumentar o desempenho e maximizar os resultados das operações em um SMA, é necessária uma estratégia de manufatura para controlar os diversos níveis existentes no SMA. Para isso é necessário um planejamento estratégico que envolve sistemas de controle de sistemas de manufatura. Esses sistemas de controle utilizam tanto informações globais quanto locais do SMA a ser controlado (AGUIRRE, 2007) (MONTEVECHI et al., 2007).

Com o objetivo de se obter um SMA com melhor desempenho este deve ter um alto grau de flexibilidade e controle (MONTEVECHI et al., 2007). Uma breve definição de elementos de controle é descrita a seguir:

- Controlador inteligente;
- Controlador de chão de fábrica; (SFC – *shop floor controller*);
- Controlador de estação de trabalho (WC – *Workstation controller*);
- Controlador de equipamento (EC – *equipment controller*).

Dependendo do uso e arranjo desses elementos diferentes arquiteturas de controle podem ser formadas, as arquiteturas básicas são as seguintes:

- Centralizada;
- Hierárquica;
- Heterárquica;
- Hibrida.

Existem outras formas de abordagens do controle de chão de fábrica em SMA, assim como o uso de sistemas multiagente, sistemas holônicos de manufatura, entre outros.

2.2.3 Sistemas Flexíveis de Manufatura

Um Sistema Flexível de Manufatura (*Flexible Manufacturing System* – FMS) consiste em um grupo de estações de processamento, normalmente máquinas-ferramentas CNC (*Computer Numeric Control* ou em Português Controle Numérico Computadorizado), ou seja, é um controlador numérico que permite o controle simultâneo de vários eixos, através de uma lista de movimentos escrita num código específico. Esse grupo de máquinas é interligado por um sistema de movimentação e armazenagem automatizado e controlado por um sistema de controle por computador. Um FMS é capaz de processar uma variedade de produtos simultaneamente, porém essa variedade se limita à flexibilidade que pode ser incorporada no FMS (GROOVER, 2007).

Normalmente, em um FMS, as estações de processamento são agrupadas por setores, esses pequenos grupos de estações de processamento podem ser chamados de Células Flexíveis de Manufatura (*Flexible Manufacturing Cells* – FMC), que geralmente são grupos de elementos de um FMS nos quais são fabricados produtos similares, em outras palavras, produtos que não são idênticos entre eles, mas que passam por mesmas máquinas para serem produzidos em algum momento do roteiro de produção. Um FMS é sempre um SMA, porém um SMA nem sempre é um FMS, ou seja, em alguns casos, podem existir tanto sistemas ou células de manufatura automatizadas que produzem apenas um tipo de produto, por tanto esses não são flexíveis. (GROOVER, 2007).

A vantagem dos FMS em relação aos outros tipos de Sistemas de manufatura é sua flexibilidade que aumenta consideravelmente o desempenho em relação aos outros, pelo fato de produzir mais de um tipo de produto simultaneamente usando poucos recursos. Porém, essa flexibilidade pode trazer muitos problemas no planejamento e implementação, por exemplo, o controlador da produção deve ser planejado e programado considerando todos os tipos de produtos que o sistema pode fabricar, os tipos de produtos que cada máquina pode fabricar, todas as funções de cada elemento do sistema, todos os possíveis roteiros de produção que cada tipo de produto pode tomar, entre outros.

2.2.4 Métodos de modelagem

Para que um SMA possa ser representado, planejado, analisado e simulado existem algumas formas de modelagem e representação (MONTEVECHI et al., 2007):

- Redes de Petri;
- Cadeias de Markov;
- Teoria das Filas;
- Processos Semi-Markovianos Generalizados (GSMP) e Simulação;
- Álgebra de Processos;
- Álgebra Max-Plus;
- Lógica Temporal de Tempo Real;
- Teoria de linguagens e Autômatos.

Entre essas formas de modelagem, redes de Petri são mais utilizadas pelas abordagens, pois conseguem representar vários aspectos e características de um SMA (FANTI e ZHOU, 2002).

2.3 Definições de redes de Petri

Redes de Petri é uma técnica matemática com descrição gráfica criada para modelagem de sistemas baseados em eventos discretos que têm a capacidade de descrever as sequências de eventos, a alocação de recursos, entre outros (MURATA, 1989; FANTI e ZHOU, 2002; OLIVEIRA JUNIOR, 2006; AGUIRRE, 2007). Um grande número de pesquisas tem apresentado novas estratégias de resolução de deadlock em SMA usando redes de Petri.

Existem várias extensões de Rede de Petri com definições distintas, das quais algumas serão apresentadas mais à frente neste capítulo. A definição apresentada a seguir é a mais comum entre as várias definições existentes.

Uma definição formal de Rede de Petri (MURATA, 1989) é fornecida pela quíntupla PN = (P, T, F, W, M0), onde:

- P = {p1, p2,..., pm} é o conjunto finito de lugares (*Places*);
- T = {t1, t2,..., tn} é o conjunto finito de transições (*Transitions*);
- $F \subseteq (P \times T) \cup (T \times P)$ é um conjunto de arcos (*Arcs*);
- $W : F \rightarrow \{1,2,3,...\}$ é a função de ponderação;
- $M0 : P \rightarrow \{0,1,2,3,...\}$ é a marcação inicial;
- $P \cap T = \emptyset$ e $P \cup T \neq \emptyset$

Uma descrição gráfica de rede de Petri pode ser definida como um grafo direcionado, ponderado e bipartido consistindo de dois tipos de nós denominados lugares e transições, onde arcos podem ser direcionados de um lugar para uma transição ou de uma transição para um lugar.

A Tabela 1 apresenta uma representação visual dos elementos da Rede de Petri.

Tabela 1: Elementos da rede de Petri (MURATA, 1989)

Elemento Visual	Descrição
▯	Transição (*t*)
O	Lugar (*p*)
→	Arco
•	Marca

As regras de disparo de uma transição compõem a base da teoria de rede de Petri para simular seu comportamento dinâmico. De acordo com estas regras, uma transição t está habilitada (apta ao disparo) se todo lugar de entrada pi tiver ao menos o número de marcas w(pi,t). Uma transição habilitada pode ou não ser disparada. No disparo da transição t, de cada lugar de entrada pi, remove-se w(pi,t) marcas e, a cada lugar de saída pj, adiciona-se w(t, pj) marcas (MURATA, 1989). Um exemplo de mudança de estados através da regra de disparo é apresentado na Figura 2.

O disparo da transição t1 de PN faz com que esta mude de estado, passando da marcação inicial M0 = {1,0} para M1 = {0, 1}. O disparo de t1 gera uma nova condição que habilita t2, cujo disparo gera a marcação M3 = {1,0}. M3, por sua vez, é igual a M0, condição em que já se sabe que t1 está habilitada.

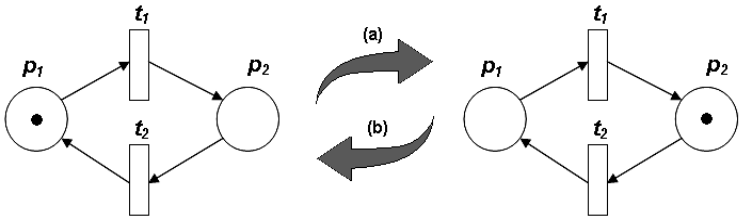


Figura 2: Mudança de estados de uma rede de Petri: (a) pelo disparo da transição t1; (b) pelo disparo da transição t2 (MAGGIO, 2005).

Eventualmente, em outras estruturas de rede de Petri com as respectivas marcações iniciais, pode se chegar a marcações com mais de uma transição habilitada, sendo arbitrária a escolha da ordem de disparo. Por outro lado, pode-se alcançar uma marcação na qual todas as transições encontram-se desabilitadas – o chamado dead-end.

Tem-se, também, o chamado conflito sempre que, na existência de duas transições habilitadas, o disparo de uma desabilita a outra, a Figura 3 apresenta um exemplo dado por Murata (MURATA, 1989) onde há uma estrutura de rede de Petri onde acontece um conflito. Neste caso, apenas uma dessas transições é disparada, escolhida arbitrariamente desabilitando a outra transição (MURATA, 1989).

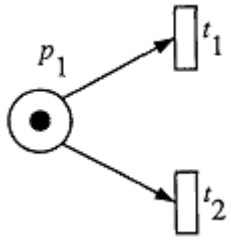


Figura 3: Uma estrutura de rede de Petri chamada de conflito. Esta estrutura apresenta um não determinismo (MURATA, 1989).

Um problema modelado em uma rede de Petri pode ser simulado através de uma sequência finita de disparos de transições, o que permite observar o comportamento do sistema ao longo das mudanças de marcação.

Existem diferentes métodos que permitem uma análise do modelo, estes são apresentados a seguir.

2.3.1 Extensões de redes de Petri

Buscando tornar o poder de modelagem mais abrangente em relação às características do sistema modelado, novas extensões de redes de Petri têm sido propostas a partir do conceito fundamental descrito anteriormente. Dentre outras extensões, estão as que incluem temporização, hierarquização, modularidade e estruturação em alto nível. A

seguir serão apresentadas as extensões denominadas redes de Petri Coloridas e redes de Petri Temporizadas, as quais, são importantes para o entendimento do projeto de pesquisa.

2.3.2 Rede de Petri Colorida

As redes de Petri Coloridas (*Coloured Petri Net - CPN*) possibilitam, como uma de suas principais características, a modelagem de diferentes tipos de marcas. Essas marcas assumem diferentes tipos de valores que são denominadas *cores* e indicam tipos distintos de dados. Dessa forma, para sistemas que apresentam subsistemas similares, a modelagem pode ficar mais compacta, por representar diferentes processos ou recursos em uma mesma sub-rede (OLIVEIRA JUNIOR, 2006; AGUIRRE, 2007).

Formalmente, uma CPN (JENSEN, F. V. et al., 1992) é definida por:

- CPN = (Σ, P, T, A, N, C, G, E, I) onde:
- Σ é um conjunto finito e não vazio de cores;
- P é um conjunto finito de lugares (*Places*);
- T é um conjunto finito de transições (*Transitions*);
- A é um conjunto finito de arcos, tal que $P \cap T = P \cap A = T \cap A = \emptyset$ (*Arcs*).
- N: A→ P x T ∪ T x P é a função nó (*Knots*).
- C: P→ Σ é a função cor.
- G: T→ EXPR é a função de guarda, onde EXPR é um conjunto de expressões. A função de guarda associa cada transição da rede a uma expressão do tipo booleana na qual as variáveis da expressão devem possuir tipos pertencentes ao conjunto de cores.
- E: A→ EXPR é a função de arco, onde EXPR é um conjunto de expressões. A função de arco associa cada arco de rede a uma expressão onde cada expressão de arco é avaliada sobre um conjunto de cores.
- I: P→ EXPR é a função de inicialização, onde EXPR é um conjunto de expressões. A função de inicialização mapeia cada lugar da rede a uma expressão, onde a cada expressão possui tipos pertencentes ao conjunto de tipos da rede.

2.3.3 Rede de Petri Temporizada

Este tipo particular de rede de Petri (*PN*) possibilita a inserção de novos atributos, aqui relacionados ao fator tempo, ao modelo de representação o que permite, além das

análises já discutidas, uma avaliação em relação ao desempenho do sistema. A variação proposta implica tanto em sua forma de representação gráfica quanto em seu formalismo descritivo.

A temporalidade em uma *PN* pode ser estabelecida associando o atraso às transições ou aos lugares da rede. Uma rede é dita P-temporizada quando o tempo é associado exclusivamente nos lugares e temporizada, quando o tempo é associado exclusivamente às transições. Isso significa que para o primeiro caso, a marca levará um determinado tempo para estar disponível definindo o atraso de habilitação da transição, já no segundo caso, para as redes T-temporizadas, a marcação fica disponível no tempo imediato, porém o disparo da transição leva um determinado tempo (MAGGIO, 2005; OLIVEIRA JUNIOR, 2006; AGUIRRE, 2007).

- A definição formal de uma rede de Petri Temporizada é definida por $TPN = (P, T, F, W, M_0, \delta)$ onde os elementos P, T, F, W, M_0 seguem as mesmas descrições de uma *PN* e δ: P $\rightarrow$ [0, +∞] é a função de atraso associado aos lugares ou δ: T $\rightarrow$ [0, +∞] a função de atraso associado às transições. $\delta(P_i) = d_i$, onde $d_i \in$ [0, +∞] e $P_i \in$ P e $\delta(t_i) = d_i$, onde $d_i \in$ [0, +∞] e $t_i \in$ T, respectivamente (AGUIRRE, 2007).

2.3.4 Propriedades Comportamentais em redes de Petri

A aplicação de rede de Petri não se restringe meramente a fins descritivos, pois permite o uso de métodos para inferir no modelo propriedades que ajudam a identificar características desejáveis e indesejáveis no sistema modelado (MURATA, 1989).

Uma das vantagens das redes de Petri é a possibilidade de análises de várias propriedades e problemas associados ao sistema modelado. Segundo Murata (1989) existem dois tipos de propriedades comportamentais que podem ser estudadas nos modelos de rede de Petri: **propriedades dinâmicas** que dependem da marcação inicial (*making-dependent*) e **propriedades estruturais** que não dependem da marcação inicial.

A seguir, serão apresentadas algumas propriedades, que serão abordadas posteriormente neste documento.

Reachability (Alcance)

Essa é uma base fundamental para o estudo das propriedades dinâmicas de qualquer sistema. O disparo de uma transição habilitada vai distribuir novas marcações através dos nós da rede de acordo com as regras de disparo descrita no Item 2.3. O

conjunto de todas as possíveis marcações que são alcançadas a partir da marcação inicial é alcançável.

Em alguns casos, o conjunto inicial de marcações não chega a todos os *places* existentes no modelo de rede de Petri, por tanto, esse conjunto de *places* não são alcançáveis.

Boundeeness (Limitação)

Uma rede de Petri é dita Limitada-simples (*k-bounded* ou *simply bounded*) se o número de marcações em cada *place* não excede um número finito (k) de qualquer marcação alcançável para todos os *places*.

Uma rede de Petri é dita segura (*safe*) se essa é Limitada-1 (*1-bounded*). Por exemplo, os modelos de rede de Petri apresentados na Figura 4 e Figura 5 são limitados, porém, o Modelo da Figura 4, em particular, é limitado-2 sendo que o da Figura 5 é seguro.

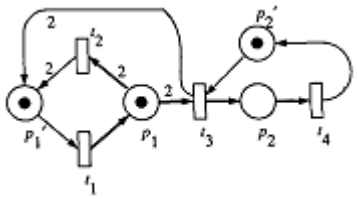


Figura 4: Modelo de rede de Petri 2-bounded (MURATA, 1989).

Os *places* em redes de Petri são muito usados para representar *buffers*, e registros para armazenamentos de informações temporárias ou intermediárias. Para verificar se a rede é limitada ou segura é necessário, esta deve garantir que não ocorrerá sobrecarga nos *buffers* ou registros, não importando qual será a sequência de disparo.

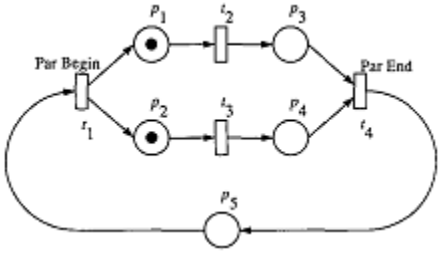


Figura 5: Modelo de rede de Petri representando atividades paralelas determinísticas (MURATA, 1989).

Liveness (Vivacidade)

O conceito de vivacidade é rigorosamente relacionado à total ausência de *deadlock* em sistemas operacionais. Uma rede de Petri é dita viva se, qualquer marcação alcança o conjunto das marcações iniciais, ou seja, é possível disparar qualquer transição da rede

passando por uma sequência de futuras transições. Isto significa que uma rede de Petri viva garante operações livre de *deadlock*. Um exemplo de uma rede de Petri viva é mostrado na Figura 5. Por outro lado, o modelo de rede de Petri mostrado na Figura 6 não é vivo já que nenhuma transição pode disparar se t_1 disparar primeiro.

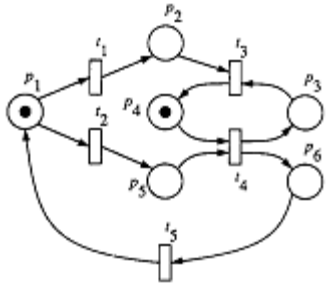


Figura 6: Exemplo de um modelo de rede de Petri segura e não viva, (L1-live) (MURATA, 1989).

Vivacidade é uma propriedade ideal para vários sistemas, pois uma verificação rigorosa dessa propriedade pode ser impraticável e muito custosa para alguns sistemas complexos como sistemas operacionais, por exemplo. Deste modo, em Murata (1989) são definidos diferentes níveis de vivacidade da seguinte forma:

- *L0-live* (*dead*) – Se *t* nunca irá disparar qualquer sequência de disparo em $L(M_0)$, ou seja, nenhuma marcação irá retornar ao estado inicial da rede para repetir a sequência de disparos;
- *L1-live* (*Potentially Firable*) – Se *t* pode disparar pelo menos uma vez a sequência de disparos em $L(M_0)$.
- *L2-live* – Se, dado qualquer inteiro positivo k, t pode ser disparado pelo menos k vezes em alguma seqüência de disparo em $L(M_0)$;
- *L3-live* – Se t aparece infinitamente, muitas vezes, em alguma seqüência de disparo;
- *L4-live* ou *live*– Se t é *L1-live* por cada marcação M em R (M0);

Lk-live – Se todas as transições da rede são *Lk-live*, sendo k = 0, 1, 2, 3, 4. L4 *liveness* é o mais forte e corresponde à vivacidade definido anteriormente. É fácil ver as seguintes implicações: L4 *liveness* -> *liveness*-L3 -> L2 *liveness* -> L1 *liveness*, onde - significa> "implica". Dizemos que uma transição é estritamente Lc vivo se for LK-vivo, mas não L (k +1)-live, para k = 1, 2, 3.

Reversibility (Reversibilidade)

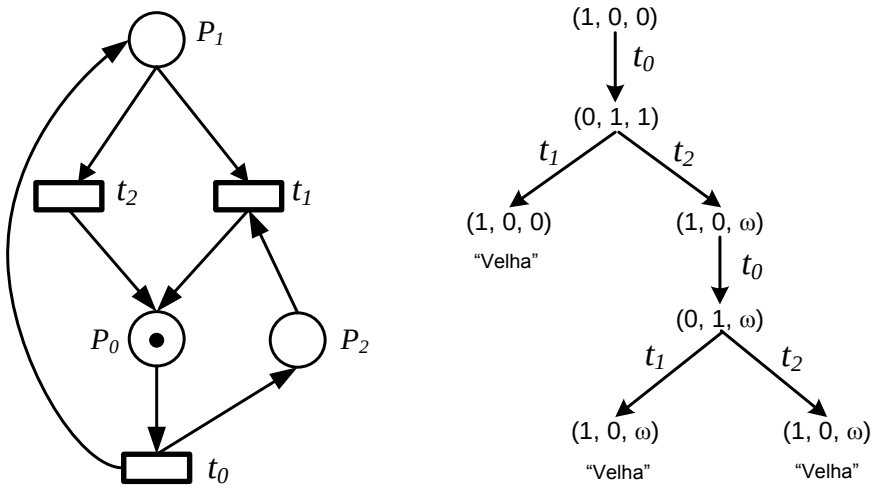
Uma rede de Petri é dita reversível se, as marcações iniciais sempre conseguirem voltar aos seus estados iniciais (*Home State*) após a sequência de disparos. Em várias aplicações não é necessário que as marcações voltem ao estado inicial.

2.3.5 Métodos de análises de redes de Petri

Os métodos de análises permitem uma melhor investigação das propriedades do modelo de maneira a examinar suas características. Os métodos são divididos em três tipos.

- Árvore de Cobertura
- Equação de estados
- Regras de Redução

O primeiro é baseado na construção de uma árvore representando todas as possíveis marcações a serem atingidas. Isso significa que as marcações existentes no modelo são enumeradas a partir de M_0 até que não existam novas marcações. O processo consiste na execução de um algoritmo que explora exaustivamente as possíveis marcações decorrentes do disparo das transições do modelo. Para uma rede limitada, a árvore de cobertura é denominada árvore de alcançabilidade, pois explora todos os possíveis estados do modelo. Do contrário, para uma rede de infinitos estados, é utilizado o símbolo ω nas marcações, indicando um inteiro positivo arbitrariamente grande. A Figura 7 ilustra esse princípio, onde a raiz da árvore é composta pela marcação inicial, e o arco indica as transições que foram disparadas originando novas marcações.



(a) Modelo Fictício (b) Árvore de Cobertura

Figura 7: Exemplo de construção da Árvore de Cobertura - adaptado de Oliveira Junior (2006)

O método de equação de estados utiliza equações algébricas simples de maneira a determinar as propriedades da rede. E o terceiro método é baseado na técnica de redução dos elementos gráficos como forma de simplificar o modelo (MURATA, 1989).

2.3.6 Softwares de modelagem de redes de Petri

Assim como um arquiteto utiliza ferramentas e softwares para modelar plantas de construções para facilitar a visualização e identificar possíveis erros de planejamentos permitindo assim uma correção antes da construção iniciar. Existem ferramentas para a modelagem de redes de Petri que permitem não só a visualização, mas também usam alguns métodos de análises e simulam comportamentos da rede de Petri (JENSEN, K. e KRISTENSEN, 2009).

Jensen (2009) propôs juntamente com a extensão de rede de Petri Colorida, um software de modelagem gráfica e matemática chamado CPN Tools. Esta ferramenta prática permite a modelagem, visualização, análises das propriedades comportamentais das redes de Petri e a validação das mesmas. CPN Tools foi a ferramenta utilizada neste trabalho parra a modelagem e simulação do método proposto. Um tutorial básico sobre como utilizar a ferramenta é apresentado no apêndice deste documento.

Outra ferramenta muito usada por propostas encontradas na literatura é o chamado Integrated Net Analyzer (UZAM e ZHOU, 2007a, 2007b; LI e SHPITALNI, 2009) que permite a análise e validação dos modelos de Rede de Petri ordinárias e coloridas (INA, 2003).

Capítulo 3

TRABALHOS RELACIONADOS

3.1 Considerações Iniciais

Sistemas de Manufatura Automatizados (SMA) diferenciam-se dos outros sistemas de manufatura por algumas características como a automatização no processo de produção, ou seja, em setores como o transporte de matéria-prima e o controle da produção, outra diferença na caracterização de um SMA é o compartilhamento de recursos, onde recursos são usados em diferentes etapas do processo de produção. De uma maneira geral, existem dois tipos de recursos: preemptíveis e não preemptíveis. Um recurso preemptível pode ser retirado de um processo ou tarefa que esteja em posse do mesmo sem que ocorra qualquer falha. Um recurso não preemptível não pode ser retirado do processo ou tarefa que o têm sem que ocorra alguma falha (TANENBAUM, 1992).

O compartilhamento de recursos em SMA pode dar condições para a ocorrência de *deadlocks.*

Deadlock, do inglês bloqueio ou impasse, pode ser formalmente definido como uma situação na qual um conjunto de tarefas ou processos estiver esperando por um evento que somente outro processo desse mesmo conjunto poderá fazer acontecer (TANENBAUM, 1992). Nesse documento a expressão será mantida no inglês.

Essa situação de *deadlock* pode ocorrer por causa das seguintes condições (COFFMAN et al., 1971):

1. Tarefas requerem o controle exclusivo dos recursos necessários ("condição de exclusão mútua");
2. Tarefas que detêm recursos já alocados, enquanto esperam por recursos adicionais (“condição de posse e espera”);

3. Os recursos não podem ser removidos forçadamente de tarefas que os detêm até que os recursos sejam utilizados por completo ("condição de não preempção");
4. Existe uma cadeia circular de tarefas, de tal forma que cada tarefa possui um ou mais recursos que estão sendo solicitados pela tarefa seguinte da cadeia ("condição de espera circular").

Na Figura 8 é apresentado um exemplo de alocação de recursos através de grafos dirigidos proposto por Holt (1972). Nesse modelo existem dois tipos de nós: processos simbolizados na figura como círculos e recursos simbolizados na figura como quadrados.

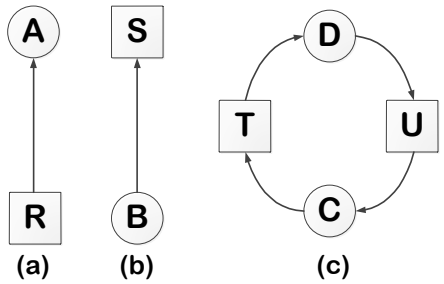


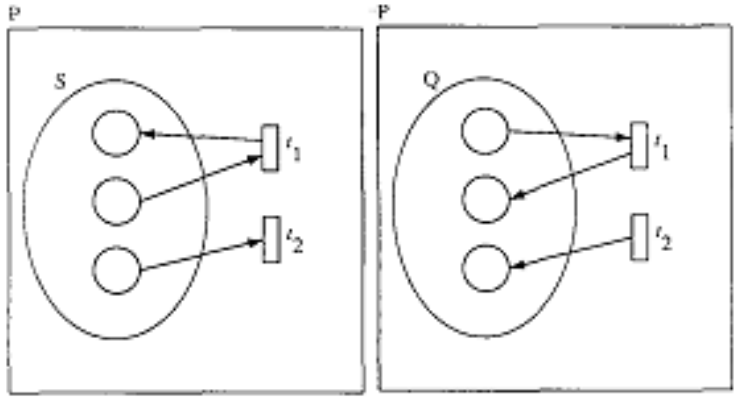
Figura 8: Grafo de alocação de recursos. (a) Processo de posse de um recurso. (b) Processo requisitando um recurso. (c) Deadlock. - adaptado de (TANENBAUM, 1992)

O arco que parte de um recurso para um processo (a) indica que o recurso foi previamente requisitado e o arco que parte de um processo para um recurso indica que o recurso requisitado está alocado ao processo (b). Na c, é apresentado um exemplo de uma situação de *deadlock*, na qual o processo *C* está esperando pelo recurso *T*, o qual está atualmente sendo usado pelo processo *D*. O processo *D* não se encontra prestes a liberar o recurso T, pois ele espera pelo recurso *U*, usado por *C*. Ambos os processos vão esperar para sempre.

Outro exemplo é proposto por Murata (1999), que define as regiões críticas onde ocorrem *deadlocks* como sifões e laços (*traps*):

Segundo Murata (1999), um subconjunto de *places* não vazio S em uma rede ordinária N consiste em um *deadlock*, ou seja, todas as transições que tem um *place* de saída em S têm um *place* de entrada em S (Figura 9a). Um subconjunto não vazio Q em uma rede ordinária N é chamado de laço (Figura 9b), ou seja, toda as transições tendo um *place* de entrada em Q têm um *place* de saída em Q. Um sifão é ilustrado na Figura 9a, onde a contagem de marcações no sifão permanece a mesma através do disparo de t_1, mas diminui pelo disparo de t_2. Assim, um sifão tem propriedade comportamental que se ele é livre de marcações no âmbito de cada marcação, então ele permanece livre de marcações no âmbito de cada marcação sucessora. Um laço é ilustrado na Figura 9b, onde a contagem

de marcações no laço permanece a mesma pelo disparo de t_1, mas aumenta através do disparo de t_2. Assim, um laço tem uma propriedade comportamental que se ele é marcado (ou seja, se ele tem pelo menos uma marcação) no âmbito de qualquer marcação. Isto é fácil de verificar porque a união de dois sifões (laços) é novamente um sifão.



$\bullet S = \{t_1\}$ $\quad$ $Q\bullet = \{t_1\}$

$S\bullet = \{t_1, t_2\}$ $\quad$ $\bullet Q = \{t_1, t_2\}$

$\bullet S \subseteq S\bullet$ $\quad$ $Q\bullet \subseteq \bullet Q$

Figura 9: Ilustração de um sifão (a) e um laço (b) (MURATA, 1999)

Um sifão é chamado de sifão básico (*basic trap)* se ele não é representado como a união de outros sifões (*traps)*. Todos os sifões em uma rede de Petri podem ser gerados pela união de alguns sifões básicos. Um sifão é dito ser mínimo se ele não contém nenhum outro sifão. Um sifão mínimo (*minimal siphon*) é um sifão básico, mas nem todos os sifões básicos são mínimos (MURATA, 1999).

Por exemplo: Em uma rede de Petri mostrada na Figura 10, possui $S_1 = \{p_1, p_2, p_3\}$, $S_2 = \{p_1, p_2, p_4\}$, $S_3 = \{p_1, p_2, p_3, p_4\}$, $S_4 = \{p_2, p_3\}$ e $S_5 = \{p_2, p_3, p_4\}$. Então, tem-se $\bullet S_1 = \{t_1, t_2, t_4\} \subseteq S_1\bullet = \{t_1, t_2, t_3, t_4\}$. Assim, S_1é um sifão. Uma vez que $S_4\bullet = \{t_1, t_4\} \subseteq \bullet S_4 = \{t_1, t_2, t_4\}$, S_4 é um laço (*trap*). Similarmente, é fácil verificar que S_2 é um laço, S_3 é um sifão e um laço e S_5 é um laço. De fato, S_1 e S_2 são sifões mínimos e básicos. S_3, S_4 e S_5 são laços base (*basis traps*), S_3 e S_5 não são laços mínimos (*minimal traps*) (MURATA, 1999).

Em muitos sistemas de manufatura, o tratamento dos *deadlocks* pode ser ignorado pois os operadores poderiam resolver o problema abortando partes do processo de produção que estavam envolvidos com o *deadlock*, somente quando ocorriam, garantindo os recursos para outras partes do processo. Com o avanço da tecnologia, as estruturas dos SMA passaram a ficar mais complexas e com isso as ocorrências de *deadlock* passaram a ficar mais frequentes e a gerar custos para as empresas manufatureiras. Nos últimos anos, várias pesquisas estão sendo feitas para resolver o problema de *deadlock* em SMA (ELMEKKAWY e ELMARAGHY, 2003) no sentido de se buscar o aumento do desempenho

desses sistemas. Segundo Fanti (2002), existem três principais métodos direcionados para problemas de *deadlock* em Sistemas de Manufatura Automatizados (SMA). São eles:

- Detecção e Recuperação de *Deadlock*, que faz monitoramento do sistema com o objetivo de detectar ocorrências de *deadlock* para então executar ações predeterminadas para recuperar o sistema;
- *Deadlock* Prevention, que utiliza políticas de restrições ao modelo no planejamento do sistema de controle do SMA;
- *Deadlock* Avoidance, que utiliza o conhecimento da alocação de recursos correntes e de futuros comportamentos dos processos para controlar a alocação e liberação de recursos de forma dinâmica.

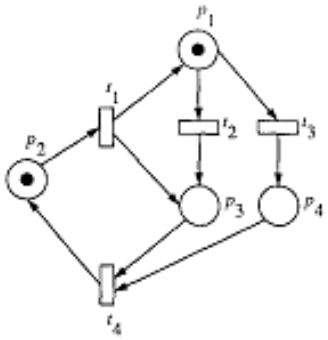


Figura 10: Exemplo de uma rede de Petri com sifões e laços (MURATA, 1999)

Em cada máquina uma peça requer um conjunto de recursos para completar seu processo de produção. Assim, tendo os SMA como um Sistema de Alocação de Recursos (*Resource Allocation System* – RAS), Reveliotis (1999) introduz a seguinte taxonomia:

1. RAS de unidade simples (*Single Unit* RAS – SU-RAS), ou seja, em cada passo do processo, uma peça requer uma simples operação de um único tipo de recurso;
2. RAS de tipo simples (*Single Type* RAS- ST-RAS), ou seja, em casa passo do processo uma peça requer várias operações de um único tipo de recurso;
3. RAS conjuntivo (AND RAS), ou seja, todos os estágios do processo requerem um número arbitrário de operações de um número arbitrário de tipos de recursos;
4. RAS Conjuntivo/Disjuntivo (AND/OR RAS), ou seja, em todos os estágios do processo uma peça requer um conjunto de tipos operações (AND) e a alocação é realizada em qualquer um recurso (OR) que se adeque para sua execução (assim este modelo de RAS possui roteiros flexíveis).

A seguir são apresentadas algumas técnicas usando as três principais estratégias de resolução de *deadlock* e a taxonomia de alocação de recursos descrita por Reveliotis (1999).

3.2 Detecção e Recuperação de *deadlock*

Detecção e Recuperação de *deadlock (Deadlock Detection and* Recovery - *DR*) trata o problema somente quando o *deadlock* ocorre, por exemplo, durante o processo de produção de um SMA, o *deadlock*, uma vez ocorrido, pode ser detectado por meio de mecanismos de monitoramento (FANTI e ZHOU, 2002).

O problema pode ser solucionado por meio de preempção de recursos, ou seja, retomando provisoriamente um recurso de seu atual proprietário e passando-o a outro processo ou tarefa, em muitos casos nos SMA essa solução não é possível. Outra forma de solução é a recuperação por meio de reversão de estado, onde os estados do processo são armazenados periodicamente de forma que, se ocorrer um *deadlock* seja possível ver quais recursos são necessários para serem revertidos para um estado em um instante anterior ao momento em que se encontra em *deadlock*. Outra maneira de recuperação de *deadlock* é por meio da eliminação de processos ou tarefas, esta é a considerada a mais simples de recuperação de *deadlock*, mas também é considerada a mais grosseira, pois simplesmente elimina um processo ou tarefa que esteja envolvido com o *deadlock* liberando o recurso para os outros (MURATA, 1989).

Em um SMA, devido sua complexidade e concorrência entre recursos essa estratégia é recomendada para sistemas, nos quais raramente ocorrem *deadlock* e que a recuperação não seja tão cara nem demorada. Algumas propostas como Elmekkawy e Elmaraghy (2003) e Hsieh (2003)utilizam essa estratégia juntamente com a estratégia de *Deadlock Avoidance*, que será detalhada no próximo item, para otimizar o processo de produção considerando o *makespan*, ou seja, tempo total de produção, que é medido a partir do momento em que o primeiro produto entra no processo de produção até o momento em que o último produto tem sua produção concluída. A utilização da estratégia *DR* juntamente com *Deadlock Avoidance* (DA) utilizada em Elmekkawy e Elmaraghy (2003) tem como objetivo resolver ocorrências de *deadlock* que DA não tem capacidade de solucionar por algum evento que não pode ser previsto pela estratégia em seu escopo (FANTI e ZHOU, 2002).

3.3 Deadlock Prevention

Os métodos de prevenção de *deadlock* chamados *Deadlock Prevention (DP)* utilizam regras, políticas e restrições no planejamento do sistema. É necessário verificar pontos do processo de produção com o objetivo de detectar e fazer mudanças para que não ocorra um *deadlock* no processo, esse tipo de estratégia necessita de informações sobre os recursos do sistema de manufatura e não requer conhecimento dos estados do controle de produção em tempo de operação. Na modelagem do sistema, levando em consideração todas as informações necessárias ao seu funcionamento, é possível aplicar políticas que previnem os *deadlocks* (UZAM, 2002, 2004; HUANG, 2007; LI e SHPITALNI, 2009; PIRODDI, CORDONE et al., 2009a, 2009b;).

DP pode ser aplicado utilizando Teoria das Regiões (*Theory of Regions*) utilizando modelos de redes de Petri explorando grafo de alcançabilidade (UZAM, 2002), mas existe um problema ao aplicar essa política em redes de Petri grandes, o que normalmente é necessário ao se modelar cenários de SMA, podendo ocorrer o "problema de explosão de estados", ou seja, com a aplicação da política de DP o número de elementos do modelo de Rede de Petri aumenta de modo exponencial (UZAM, 2004). O "problema de explosão de estados" pode ser resolvido aplicando a abordagem de redução de redes de Petri (MURATA, 1989), que simplifica a estrutura conservando as propriedades, assim como limitação, vivacidade e reversibilidade (UZAM, 2004).

Em Uzam(2004) a estratégia de DP foi utilizada em um Sistema Flexível de Manufatura complexo, no qual a política utilizada tem como uma das estratégias reduzir a complexidade da modelagem em rede de Petri em pequenas sub-redes mantendo a estrutura e propriedades da modelagem original, podendo assim solucionar o problema de *deadlock* em cenários mais complexos, evitando estados em que ocorra um *deadlock*.

Em Uzam (2007) foi proposto um supervisor de execuções de vivacidade (*Liveness Enforcing Supervisor* – LES) em um FMS usando modelos de Rede de Petri na modelagem. O LES consiste em um número de lugares (*places)* de controle, unidos com seus arcos relacionados e marcações iniciais, é computado como uma rede de Petri. Em uma Rede de Petri dita viva pode existir *places* de controle redundantes. Um *place* controlado é chamado redundante se a remoção ainda mantém a rede viva. Deve-se notar que esta definição é diferente da definição de um lugar redundante. Remover este não altera o grafo de alcançabilidade. Uzam (2007) propõe um algoritmo de teste de redundância para LES em FMS, que ajuda na geração do grafo de alcançabilidade da rede. Segundo Uzam (2007) o teste de redundância é aplicável á qualquer modelo de rede de Petri desde que esta seja

viva, dentro do escopo de *deadlock* em FMS, juntamente com a implementação de *places* controlados.

Na Figura 11 É mostrado como o teste proposto por Uzam (2007) age em uma rede de Petri: Antes de o algoritmo iniciar, deve-se montar um modelo de rede de Petri e, então adicionar *places* controladores, que geram uma rede de Petri viva a partir do sistema modelado. Com o modelo de rede de Petri viva gerado, o algoritmo de teste de redundância obtém informação da rede assim como: o número de *places* da rede, o número de *places* controlados (CP), o número de *places* redundantes e o número necessário de CPs, então, são executados dois algoritmos que removem os *places* redundantes desde que, essa remoção não faça com que a rede perca sua vivacidade. O primeiro algoritmo (A) parte de um estado inicial e percorre todos os *places* alcançáveis da rede até o último, após algumas iterações definidas arbitrariamente é iniciado o algoritmo B, que a partir da rede resultante do algoritmo A executa os mesmos procedimentos, mas desta vez, o algoritmo parte na ordem inversa do algoritmo A.

Tanto o teste A quanto o teste B podem retornar um mesmo resultado ou resultados diferentes, isso depende do sistema de controle de vivacidade considerado.

Segundo Uzam (2007), o algoritmo de teste de redundância é fácil de usar e muito eficaz. Sua complexidade, no entanto, exponencial em relação ao tamanho da rede modelada, uma vez que seja necessário gerar o grafo de alcançabilidade.

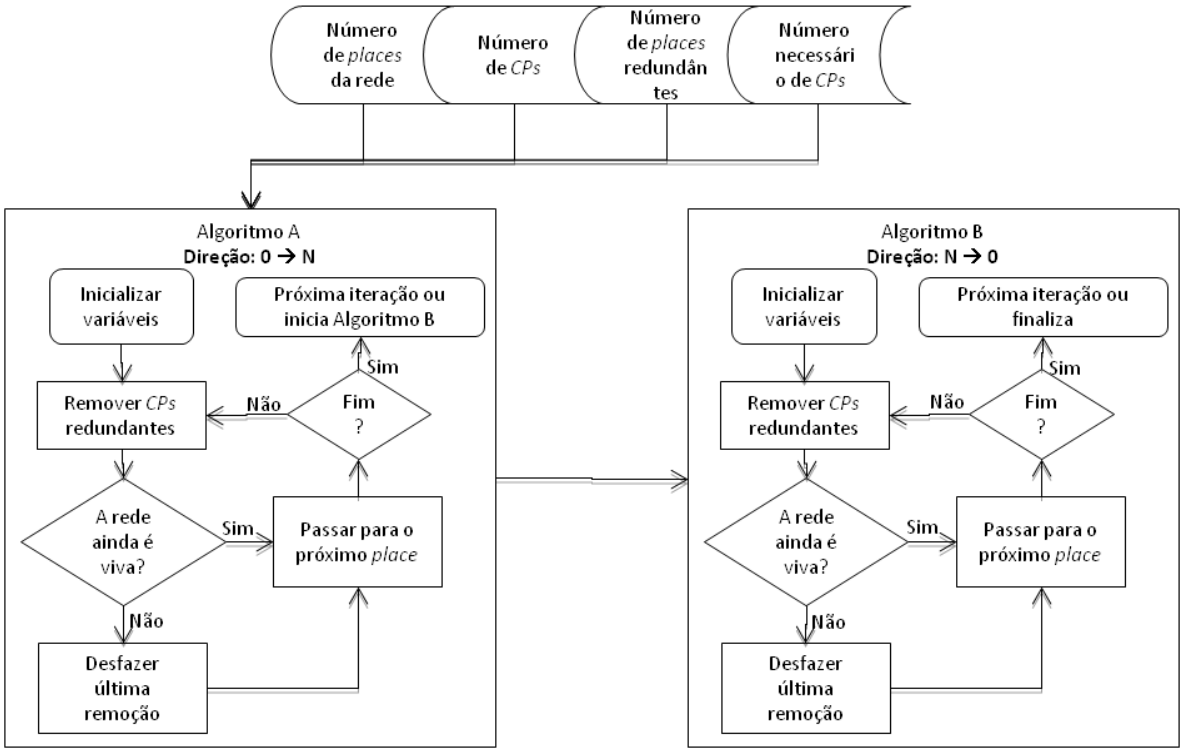


Figura 11: Teste de redundância proposto por Uzam (2007).

Após o teste de redundância um modelo de rede de Petri vivo é obtido, o teste foi aplicado em dois cenários foram encontrados na literatura, os testes apresentaram resultados satisfatórios, já que os modelos de rede de Petri se mostraram livres de *deadlock* e com certo controle do tamanho da rede.

Outro método para aplicar a estratégia de DP é aplicando regras com o uso da abordagem de controle de sifões com supervisão na vivacidade do modelo de rede de Petri (HUANG, 2007). Essa abordagem trata o problema de *deadlock* causado por sifões vazios em uma rede de Petri modificada, onde os sifões podem ser analisados mais facilmente e tornando o algoritmo de controle mais permissivo, ou seja, o algoritmo de controle permite mais sincronismo e paralelismo no sistema, deixando mais recursos sendo utilizados ao mesmo tempo durante o processo de produção, resultando assim, em um maior desempenho no sistema.

A Proposta de Huang (2007) usa um modelo de supervisor modelado em S^3PR (uma variação de rede de Petri proposta por Ezpeleta, Joaquin et al. (1995)) em um Sistema Flexível de Manufatura. Nesse cenário foi proposta uma política de DP utilizando uma abordagem de controle de sifões com o objetivo de manter a vivacidade da rede de Petri adicionando novos *places* a transições dissipadoras de marcações (*sink transitions*) em transições que nunca recebem marcações (*souce transitions*) (MURATA, 1989), como mostra a *Figura* 12. Depois de aplicado o controle de sifões, a proposta aplica uma política de DP. A política de *DP* aplicada primeiramente minimiza o número de sifões através de um algoritmo que verifica se existem sifões que podem ser removidos garantindo a integridade da rede, após a minimização dos sifões são aplicados passos que verificam pontos no modelo que podem causar *deadlock* e assim aplica alterações preservando a vivacidade e a reversibilidade da rede.

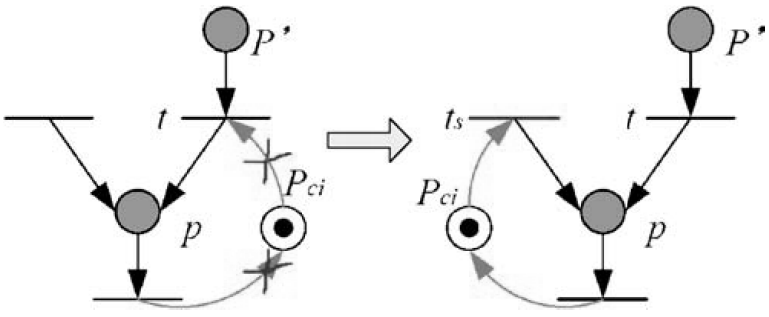


Figura* 12*: Controle de estados adicionando em uma sink transition (HUANG, 2007)

A técnica foi comparada com outras abordagens encontradas na literatura, através de dois cenários. Segundo Huang (2007), essa técnica de DP, juntamente com o controle de

sifões retornou um algoritmo de controle mais permissivo em relação às abordagens comparadas, ou seja, um controle que pode trazer um desempenho melhor no sistema.

Outra abordagem para a estratégia DP utiliza sifões seletivos, em outras palavras, a abordagem foca no problema de redundância e propõe um critério diferente para selecionar sifões não redundantes para o controle, focando principalmente o tempo de processamento do controle de produção (PIRODDI et al., 2008).

No Trabalho de Piroddi, Cossalter et al. (2009) foi proposta uma abordagem de desacoplamento de recursos para DP em Sistemas Flexíveis de Manufatura. A modelagem do fluxo de produção juntamente com os estados das máquinas utiliza um processo sequencial simples (S^2P *Simple Sequencial Process*). O método apresentado utiliza um sistema de controle baseado em sifões, que pode fazer um sistema livre de *deadlock* adicionando restrições que preserva cada sifão.

3.4 Deadlock Avoidance

Deadlock Avoidance é uma estratégia que anula uma ou mais condições necessárias para a ocorrência de *deadlock* antes de sua ocorrência, armazenando a sequência do estado atual e de possíveis condições futuras (FANTI 2004). Esta é uma abordagem dinâmica que utiliza o conhecimento da alocação de recursos correntes e de futuros comportamentos dos processos para controlar a alocação e liberação de recursos. Segundo Fanti (2002), A principal característica da estratégia de *Deadlock Avoidance* é que esta é executada em tempo de operação e toma como base para as decisões as informações dos estados dos recursos, mais precisamente, um controlador *Deadlock Avoidance* habilita ou desabilita eventos que envolvem aquisição ou liberação de recursos usando procedimentos *look-ahead* (olhar à frente) (WU, MENGCHU e ZHIWU,2008).

A seguir são apresentadas algumas técnicas de *Deadlock Avoidance.*

3.4.1 Resolução de *deadlock* com serviço de recursos conjuntivos

Esta estratégia de resolução de *deadlock* apresentada por Fanti (2004) controla a alocação dos recursos impedindo o *deadlock* em sistemas de alocação de recursos do tipo SU-RAS.

Nas regras de prioridades utilizadas em Fanti (2004), que diferente de outra proposta como Hsieh (2000)*, as tarefas são classificadas em três diferentes tipos:*

- Novas Tarefas.

- Uma tarefa “j” passa de um recurso a outro.
- Uma tarefa é completada e o recurso é liberado.

Para cada um dos dois primeiros tipos de tarefas apresentados são executadas duas ações distintas:

- Ação 1 – Identifica as regras de fluxo das tarefas no SMA
  - Verificação lógica baseada no conhecimento da próxima marca
- Ação 2 – Rege a:
  - Entrada de tarefas no sistema
  - Base de decisão na evolução de controles futuros do sistema

A terceira tarefa só é classificada com o objetivo de controlar o fluxo de produção e passar a informação que o recurso que estava sendo utilizado foi liberado.

Fanti (2004) utiliza um controlador lógico de DA que busca informações em uma base de conhecimento da próxima marcação do modelo de rede de Petri Colorida Temporizada e gerencia a entrada de novas tarefas do sistema.

Método de modelagem.

A Figura 13 apresenta um SMA com três estações de trabalho (r_1, r_2, r_3), uma estação de carga e descarga (r_4) e dois robôs de movimentação de peças (r_5, r_6), na modelagem o recurso r_7 representa a saída do sistema. Este SMA foi utilizado para exemplificar situações onde possam ocorrer *deadlock*.

Inicialmente, Fanti (2004) monta uma rede de Petri comum para usá-la como esqueleto da CTPN, como é apresentada na Figura 14.

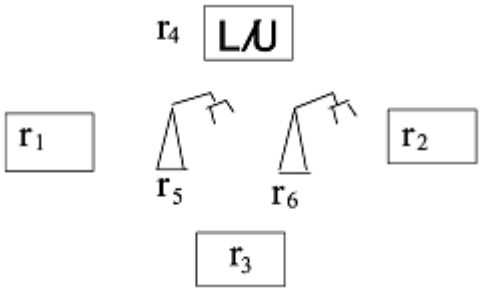


Figura 13: SMA de montagem com robôs proposto por Zhou e DiCesare(1993)

O modelo $PN = (P, T, F, H, m_0)$ apresentado na Figura 14 descreve o comportamento do SMA, onde os conjuntos de *places*, transições, arcos e arcos inibidores são representados por P, T, F e H respectivamente.

Fanti (2004) utiliza um modelo de rede de Petri Colorida Temporizada $CTPN = (P, T, C_o, H, C^+C^-, \Omega, M_0)$ que descreve o comportamento do SMA, onde P, T, H são os conjuntos de *places*, transições e arcos inibidores respectivamente. As marcações *M* representam a operação associada com as operações restantes do processo de produção do produto correspondente. Além disso, é associado um conjunto de cores a cada transição. Em especial, suponhamos que para cada marcação, o conjunto de cores correspondentes tem-se o seguinte conjunto de cores:

$$C_o(r_m) = \{< RWP, j > \ tal\ que\ j \in J\ e\ RWP$$
$$\text{é}\ uma\ subseq\text{üência}\ de\ algum\ w_k \in W$$
$$começando\ com\ qualquer\ operação\ contendo\ r_m\}.$$

Além disso C_o associa a cada transição $t_{i,i+1}^k \in T_F$ um conjunto de cores de ocorrência:

$$C_o\left(t_{i,i+1}^k\right) = \{< RWP, j > \ tal\ que\ cada\ j \in J, RWP\ \text{é}$$
$$uma\ subseq\text{üência}\ de\ algum\ w_k \in W, \alpha_i^k$$
$$pertence\ \text{à}\ primeira\ operação\ de\ RWP\ e$$
$$\rho_{i+1}^k\ pertence\ \text{à}\ primeira\ operação\ de\ RWP\}.$$

Aqui, a CTPN é representada pela matriz de incidência C para cada seta contém o *place* $r_m \in P$ e uma coluna para cada transição $t \in T$. Cada elemento é uma função que atribui um elemento $C(r_m)$ com um elemento $r_{m \in P}$. A matriz de incidência é calculada como $C = C^+ - C^-$ onde a pré e a pós-matrizes de incidência, respectivamente, são:

- • Para cada $(r_n, t) \in F, C^-(r_n, t) = I$, onde I representa "a função não faz a transformação dos elementos", Caso contrário, $C^-(r_n, t) = 0$. Esta definição significa que cada marcação deixando um recurso $r_n \in P$ não é modificada;
- • Para cada $(t, r_m) \in F, C^+(r_m, t) = U$, onde U é uma função que atualiza a cor $< RWP >$ com a cor $< RWP' >$, caso contrário, $C^+(r_m, t) = 0$. Mais precisamente, RWP'X é o procedimento de trabalho residual obtido a partir de RWP cortando seu primeiro elemento o_i.

O conjunto Ω é definida por $\Omega = \{C_o(x): x \in P \cup T\}$. Além disso, considerando que a marcação inicial M_0 sem tarefas está sendo processada, configura-se $M_0(r_n) =< 0 >$ para cada $r_n \in P$.

No entanto, na acepção CTPN cada marcação é caracterizada pela sua cor $< RWP(j), j \geq (\rho_i^k, \rho_{i+1}^k, \dots, \rho_{L_k}^k)$ e sua marca $s(< RWP(j), j >)$ que é igual ao tempo que a marcação passou no *place* atual.

• $t_{i,i+1}^k$

C1: $C(r_m) \geq |M(r_m)| - 1$;

C2: $M(r_n) \geq C^-(r_n, t_{i,i+1}^k)(< RWP(j), j >)$, ou seja, $< RWP(j), j > \in M(r_n)$.

A condição C1 refere-se à condição imposta pelos arcos inibidores e C2 representa a condição determinada pela CTPN na marcação. Se transição $t_{i,i+1}^{k} \in T_F$ satisfaz C1 e C2, diz-se habilitada por recursos e por cores, respectivamente. Além disso, pode ocorrer em $t_{i,i+1}^{k} \in T_F$ no tempo τ, se for um cor-recurso habilitada e pronta (ou seja, $s(< RWP(j), j >) \geq \tau_i^k$, onde τ_i^k é o tempo de funcionamento determinístico).

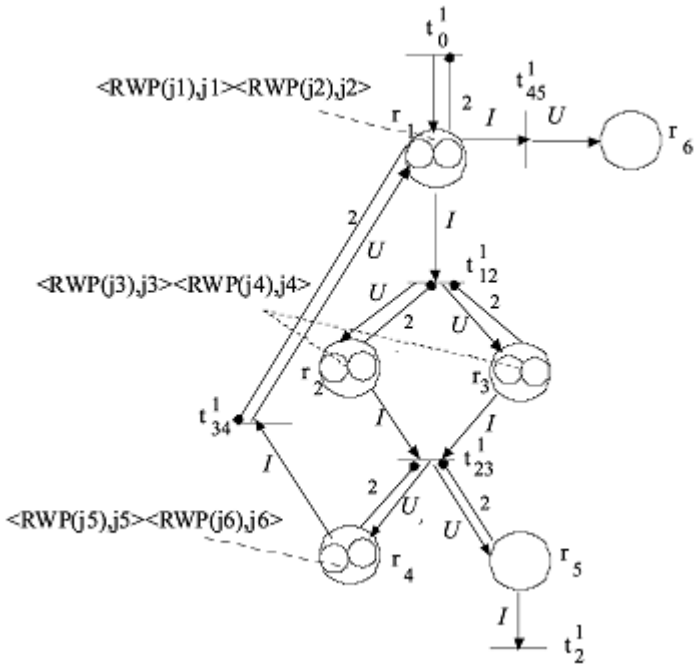


Figura 14: Modelo de rede de Petri do SMA usado por Fanti (2004)

Descrição da técnica.

Primeiramente, considerando que as ocorrências de *deadlock* dependem de uma política de gerenciamento de recursos imutável. Isto é necessário para aplicar uma estratégia de controle que evita *deadlock* e regras de alocação de recursos em cada ocorrência de evento na base de conhecimento do sistema. Em particular, os modelos de eventos discretos devem considerar dois tipos de eventos controláveis: uma nova tarefa (ordem de produção) entra no sistema (evento do tipo 1), uma tarefa sai de um conjunto de recursos e avança para o próximo (evento do tipo 2). Uma vez que um evento do tipo 3 não envolve a aquisição de recursos, este não causa *deadlock*. Consequentemente, o controlador não gerencia eventos do tipo 3. Porém, é aplicada uma política que decide se eventos do tipo 1 ou 2 podem ocorrer (controle de eventos habilitados).

Em seguida, uma especificação do mecanismo que estabelece prioridade para as tarefas correntes é necessária para evitar ambiguidade no comportamento do sistema. Por exemplo, quando mais de uma transição estão habilitadas simultaneamente, o gerenciador

do fluxo de trabalho tem que escolher qual peça será transportada para o próximo recurso. Portanto, regras de prioridade são definidas para selecionar a próxima peça que será processada. Alguns problemas concorrentes comuns são os seguintes:

1. A seleção de uma tarefa, entre as bloqueadas para receber o serviço para se tornar disponível. A tarefa escolhida será movida primeira para o próximo recurso.
2. A seleção de uma tarefa entre as que concluíram uma operação simultaneamente;
3. A seleção de uma tarefa entre as peças que têm de aquisição do próximo conjunto de recursos inibida pelo controlador. O gerenciador do fluxo de trabalho verifica se o evento associado à tarefa escolhida pode ser habilitado ou deve ser confirmado como desabilitada.

Para especificar o mecanismo de seleção acima, a política de prioridade é definida como uma tripla de funções.

$$II = (\pi_1, \pi_2, \pi_3)$$

Ou seja, $\pi_1 : M \rightarrow JM$ para $i = 1,2,3$ onde M é o conjunto de marcações admissíveis da rede de Petri Colorida Temporizada. A função π_i com $i = 1,2,3$ resolve os problemas 1,2 e 3 respectivamente.

No termo de modelagem de CTPN, uma transição $t \in T$é dita controlada se o disparo é determinado por uma política de controle (CP) quando t está pronta e habilitada de acordo com as condições C1 e C2. Assim, uma CP é uma política de restrição que determina quando uma transição é controlada e pode disparar de acordo com a CP, então a CTPN é dita controlada pela CP. Formalmente, no modelo proposto por Fanti (2004) uma CP é um mapeamento que se associa com cada evento $\sigma \in \Sigma_1 \cup \Sigma_2$ e com cada marcação *M* uma ação de controle que habilita ou desabilita evento σ.

Definição da Política de Controle (CP):

$f_i : \Sigma_i \times M \rightarrow \{0,1\}$ com $i = 1,2$

Onde $f_i(\sigma_i, M) = 0 (f_i(\sigma_i, M) = 1)$ significa que para a CTPN na marcação *M*, o evento $\sigma_i \in \Sigma_i$ com $i = 1,2$ é o controle de inibição (controle ativado).

Mais Precisamente, controlando um evento de tipo 2 $\sigma_2 = \left(j, \alpha_i^i, \rho_{+1i}^k\right)$ (ou um evento do tipo 1 $\sigma_1 = (j, w_k)$ significa que o disparo da transição correspondente $t_{i,i+1}^k$(ou t_0^k respectivamente) é controlado pela CP. Note que as transições $t_i^k \in T_R$ (correspondendo a um evento do tipo 3) e $t_{i,i+1}^k \in T_F$ com $i = L_k - 1$ para $k - 1, \dots, W$ (correspondendo ao sistema de liberação de trabalho) não requer qualquer controle porque essas ocorrências não dão condições para a ocorrência de *deadlock*.

Contudo, para evitar *deadlock* é necessário garantir que cada tarefa no processo pode ser executada somente se essas cheguem até a saída do sistema sob-controle da CP. Assim, são definidas as marcações de tarefas (*task marking*), as marcações associadas com o estado do sistema nas quais todas as tarefas do processo foram produzidas e alcançaram a saída. Mais formalmente, são introduzidas as seguintes definições.

Definição 3: A Marcação M* é chamada marcação de tarefa da CTPN se $M^*(r_R) \neq < 0 >$ e $M^*(r_n) = < 0 >$ para cada $r_n$com $r_n \neq r_R$.

Definição 4: Uma CTPN controlada pela CP é dita livre de *deadlock* na marcação $M \in Reach(M_0)$, se existir uma sequência de disparos controlados δ tais que....

Obviamente, se a CTPN possui marcações com *deadlock*, este não alcança M*. Em outras palavras, em uma marcação M* livre de *deadlock* de uma CTPN deve ser alcançável por todas as marcações M da CTPN. Assim o problema de síntese do controlador de *deadlock avoidance* que é considerado é encontrar para o modelo de CTPN um CP f_i com $i = 1,2$ para obter uma CTPN livre de *deadlock* em cada marcação $M \in Reach(M_0)$, ou seja, capaz de alcançar M* depois de completar a produção.

Começando pela Proposição 1, uma política de *deadlock avoidance* (chamada CP1) é definida, baseada em um procedimento look-ahead de apenas um passo. Mais precisamente, quando um evento tem que ocorrer, a política atualiza a próxima marcação M', construindo um novo grafo de transição $D_T(M')$ e inibe o evento se tal dígrafo contém um BCSS (explicar). Essa política de controle é definida como segue:

- $f_i(\sigma_i, M) = 0$ Se o dígrafo de transição $D_T(M')$, descrito pela matriz adjacente A_M,expondo uma BCSS.
- $f_i(\sigma_i, M) = 1$ caso contrário.

Embora o CP1 previne as transições que levam a um *deadlock* imediato, esta pode não evitar algumas situações chamadas *deadlock* restrito (*restricted deadlock*). Nessa situação, o sistema não está em estado de *deadlock*, mas inevitavelmente incorre o bloqueio permanente causado pela inibição do controle. No entanto, diversos autores mostraram que, se o sistema é um SU-RAS e utiliza algumas propriedades, os estados de *deadlock* restritos são evitados por uma CP que inibe apenas transições que causam *deadlock* nas próximas etapas. Por exemplo, isto acontece se a capacidade de cada recurso no SU-RAS é mais que um ou quando cada capacidade unitária dos recursos é equipada com um buffer de entrada ou saída. Pelo contrário um SMA com serviço de recursos conjuntivo não verifica esses resultados. Na verdade, nesses sistemas restritos pode ocorrer *deadlock* pela CP1 mesmo se a capacidade de cada recurso é maior que um. O exemplo a seguir esclarece essas considerações.

Exemplo: Considerando o sistema de cinco recursos do exemplo com capacidades $C(r_i) = 2$ para $i = 1,2,\ldots,5$. As tarefas podem ser produzidas pelo seguinte procedimento de trabalho:

$w_1 = (o_1^1, o_2^1, o_3^1, o_4^1) = ((\rho_1^1, \tau_1^1), (\rho_2^1, \tau_2^1), (\rho_3^1, \tau_3^1), (\rho_4^1, 0))$,

$w_2 = \left((\rho_1^2, \tau_1^2), (\rho_2^2, \tau_2^2), (\rho_3^2, \tau_3^2), (\rho_4^2, \tau_4^2), \left(\rho_5^2, 0\right)\right)$e

$w_3 = ((\rho_1^3, \tau_1^3), (\rho_2^3, \tau_2^3), (\rho_3^3, \tau_3^3), (\rho_4^3, \tau_4^3), \left(\rho_5^3, 0\right))$.

Supõe-se que o sistema é a marcação M com

$$J_M = \{j_i, i = 1, \dots, 5\}$$

$M(r_1) =< RWP(j_1), j_1 >< RWP(j_2), j_2 > =< \left(o_1^3, o_2^3, o_3^3, o_4^3, o_5^3\right), j_1 ><$
$\left(o_1^3, o_2^3, o_3^3, o_4^3, o_5^3\right), j_2 >;$

$M(r_2) =< RWP(j_5), j_5 > =< (o_1^1, o_2^1, o_3^1, o_4^1), j_5 >;$

$M(r_3) =< RWP(j_3), j_3 >< RWP(j_4), j_4 > =< \left(o_1^2, o_2^2, o_3^2, o_4^2, o_5^2, j_3\right) ><$
$\left(o_1^2, o_2^2, o_3^2, o_4^2, o_5^2\right), j_4 >;$

$M(r_n) =< 0 > para\ n = 4,5,6.$

Mais precisamente, o recurso r_1 está *ocupado* e retém j_1 e j_2 que requer r_2 e r_3, em sucessão. Além disso, j_3 e j_4 retém o recurso *ocupado* r_3 e requer r_2 e r_1 em sequência. Finalmente, j_5 retém r_2 e requer o serviço conjuntivo de r_1 e r_3. Agora, o evento $\sigma_2 =< j_5, r_2, \{r_3, r_1\} >$ é bloqueado porque r_3 e r_1 estão ocupados e t_{12} não está com o recurso habilitado. Além disso, CP1 inibe os eventos $< j_1, r_1, r_2 >$ e $< j_4, r_1, r_2 >$ porque eles determinam um *deadlock*, detectado pelo BCSS $\Gamma = (\{r_2, r_3\}, \{e_{32}, e_{23}\})$. Além disso, os eventos $< j_3, r_3, r_2 >$ e $< j_4, r_3, r_2 >$ são inibidos porque estes determinam um estado de deadlock detectado pelo BCSS $\Gamma' = (\{r_2, r_1\}, \{e_{12}, e_{21}\})$. Assim o estado descrito representa um deadlock restrito.

As considerações acima necessitam de uma nova CP com o objetivo de obter um comportamento livre de *deadlock* e *deadlock* restrito da CTPN.

Definição 5: A política de controle CP1* é chamada de Política de Controle Modificada obtida pela CPI se esta é procedente como segue:

CP1*:

$$f_1^*: \Sigma_1 \times M \to \{0\}$$

$$f_2^* = f_2$$

Em outras palavras, CP1* previne cada tarefa de entrar no sistema e controlar cada evento do tipo 2 como CP.

Definição 6: A marcação M* é dita alcançável pela marcação $M \in Reach(M_0)$ pela CP1* e a regra de prioridade II, se existe uma sequência de disparos controlados $\delta = t_1(c_1)t_2(c_2) \dots t_n c_n$com $t_1(c_i) \in T_F \cup T_R$ para $i = 1, \dots n$ tais que $M[\delta > M^*$.

Destaca-se que a sequência de disparos δ controlada pela CPI* é uma sucessão de eventos dos tipos 2 e 3, mas simultaneamente pode ocorrer entre as transições prontas e habilitadas. No sistema, simultaneamente é resolvido pela regra de prioridade II onde cada marcação seleciona apenas um evento para ser disparado, sem ambiguidade. Portanto, se M* é alcançável por M pela CPI* e II, a marcação M* pode ser alcançável por M sob uma regra de prioridades diferente.

Considerando a definição anterior, CP1 é modificada como segue:

CP2: Tendo a CTPN no tempo τ e na marcação $M \in Reach(M_0)$ e denotando com M' a marcação obtida por M após $\sigma_1 \in \Sigma_1$ou $\sigma_2 \in \Sigma_2$ a ocorrência dos eventos:

- $f_1(\sigma_1, M) = 1$ se M* é alcançável por M' pela CP1* e a regra de prioridade II.
- $f_1(\sigma_1, M) = 0$ caso contrário.
- $f_2(\sigma_2, M) = 0$ se o dígrafo de transição associado à M' e descrito pela matriz adjacente A_M, expõe uma BCSS.
- $f_2(\sigma_2, M) = 1$ caso contrário.

A proposição a seguir prova que o sistema controlado pela CP2 é livre de *deadlocks* e *deadlocks* restritos.

Proposição: Tem-se a CTPN no tempo τ e na marcação $M \in Reach(M_0)$. Se M* é alcançável pela marcação M pela CP1* e regra de prioridade II, então a CTPN controlada por CP2 e a regra de prioridade II é livre de *deadlock*.

Prova: Supõe-se que a CTPN no tempo τ e na marcação $M \in Reach(M_0)$. Se M* é alcançável pela marcação M pela CP1* e regra de prioridade II, então existe apenas uma sequência de disparos controlada δ assim que $M[\delta > M^*$pela CP1* e regra de prioridade II. Portanto, a evolução da CTPN controlada pela CP2 segue a sequência de disparos δ até um evento do tipo $1\sigma_1$tem de ocorrer. Além disso, supõe-se que a CTPN alcança a marcação M_1tais que $M[\delta_1 > M_1$onde δ_1é uma subsequência de δ. Chama-se M' a marcação alcançada após a ocorrência de σ_1, se existe uma nova sequencia de disparos controlada $\delta_2$pela CP1* e pela regra de prioridade II, de modo que $M'_1[\delta_2 > M^*$, então f_1 habilita σ_1, se existir uma nova sequência de disparos controlada δ_2 enquanto um novo evento do tipo 1 tende ocorrer. Concluindo, o sistema livre de *deadlock* é garantido.

As proposições apresentadas estabelecem a vivacidade (*liveness*) da rede de Petri modelando o sistema por meios de verificação de acessibilidade da marcação M*, pela política de controle e um procedimento de teste de validação suficiente que checa se as ações do controle genérico mantêm a vivacidade da rede de Petri. Pelo contrário, a proposição refere-se ao CP1* definido que inibe as transições imediatamente levando a uma marcação caracterizada como *deadlock*. Além disso, a proposição estabelece uma condição suficiente apenas para obter uma CP livre de *deadlocks* restritos.

Resultados

O método de controle de Fanti (2004) foi comparado com a política proposta de Hisieh (1994), pois as duas propostas baseiam a decisão na evolução de controle futuro do sistema. Uma distinção Da técnica proposta por Fanti (2004) é o uso da modularidade da

CTPN que facilita a atualização em tempo real do modelo, portanto, o modelo pode ser alterado seguindo as mudanças frequentes do sistema de produção.

A política de controle para evitar *deadlocks* proposta (CP2), que realiza um controle lógico baseado no conhecimento das próximas marcações da CTPN e regula as entradas de tarefas com base nos estados futuros do sistema. Além disso, a estratégia de controle proposta não exige a reconfiguração de software se houver alteração no *layout* e no processo de trabalho do SMA. Por isso, é apropriado para controlar um SMA dinâmico em que os recursos, ferramentas e *layout* podem ser alterados. Obviamente, a CP2 também pode ser aplicada a um SU-RAS. Com tudo conclui se que, a CP2 não é uma política de controle maximamente permissiva.

3.4.2 Resolução de *deadlock* usando redes de Petri orientada a recursos

Segundo Wu, Zhou e Li (2008) o problema de *Deadlock Avoidance* para Sistemas de Montagem Flexíveis (*Flexible Assembly Systems* – FAS) recebeu apenas uma atenção limitada nas últimas décadas. A proposta de Wu, Zhou e Li (2008) estuda o problema para evitar *deadlock* para a FAS com fluxo de materiais *fork/join*, também abordado por Roszkowska (1993), Roszkowska (2004). Um FAS é modelado usando um tipo especial de rede de Petri chamado rede de Petri Orientada a Recursos (ROPN), originalmente desenvolvida pelos autores em questão. Este artigo apresenta um exemplo de faz de Wu, Zhou e Li (2008) que mostra como o modelo de ROPN para FAS é desenvolvido. Baseado no modelo desenvolvido um algoritmo para calcular as necessidades de recursos realizáveis e a política de controle *Deadlock Avoidance* são apresentados. Um exemplo é usado para mostrar o desempenho da política proposta e também um estudo de caso industrial.

Método de modelagem.

Um FAS mostrado na Figura 15 é uma adaptação da proposta de Roszkowska (2004). É composto por dois robôs de r_1 e r_2, e três estações de trabalho (w_1, w_2 e w_3). Um buffer de entrada b_0 e um buffer de saída b_1 para a estação de trabalho w_1, buffers b_2 e b_3 para w_2 e w_3 respectivamente. Entre eles, b_4 pode ser acessado por w_1 e w_2, e b_5 pode ser acessado por w_2 w_3. O robô r_1 oferece bandejas com peças e conjuntos, enquanto robô r_2 move componentes de base. No processo de montagem, bandejas com peças e subconjuntos são mantidos em b_4 ou b_5, e *pallets* com componentes de base são colocados na b_0, b_1, b_2 e b_3 para a montagem. Quando uma estação de trabalho executa uma

operação, pode levar as partes ou subconjuntos em uma bandeja na b_4 ou b_5 e montá-los para os componentes de base nos buffers estação de trabalho ($b_{0\text{-}3}$).

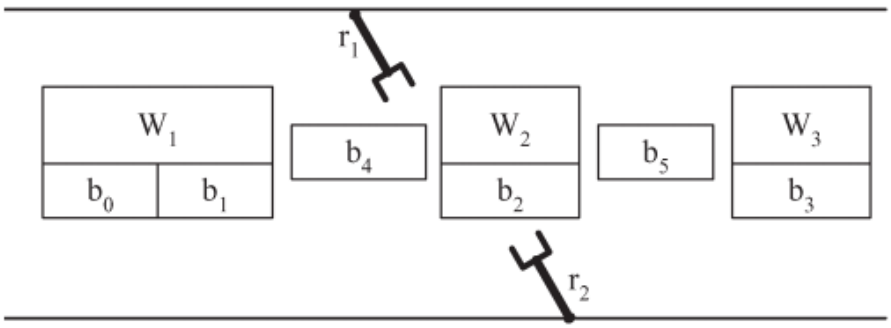


Figura 15: Cenário abordado por Wu (2008)

No exemplo, suponha que dois produtos, por exemplo, A e B, deve ser montada no sistema simultaneamente. Seus processos de montagem são mostrados na Figura 16, onde "*tray*" significa bandeja com peças e subconjuntos, e "base" significa componentes de base. Um exemplo, após robô r_1 pegar uma bandeja com peças para serem montadas sobre a base em b_4, o robô r_2 transporta um componente de base para b_0, então, w_1 monta as peças em b_4 para a base em b_0, e ao terminar é movido para b_1, enquanto a bandeja com as outras partes permanece no b_4. Então, o robô r_2 pode entregar o componente de base em b_2, e w_2 executa uma operação sobre ele. Depois disso, ele permanece em b_2. Este pode então é emitido em b_3 por r_2. Se, ao mesmo tempo, a bandeja b_4 é entregue em b_5 por r_1 e outras partes da central de armazenamento são lançados no sistema e r_1 transporta para b_5, em seguida, w_3 está pronto para realizar a sua montagem. Depois disso, o produto acabado permanece em b_3, e permanece na bandeja do *buffer* b_5, respectivamente. Eles estão prontos para ser entregues para a saída do sistema, e os espaços de reserva são liberados.

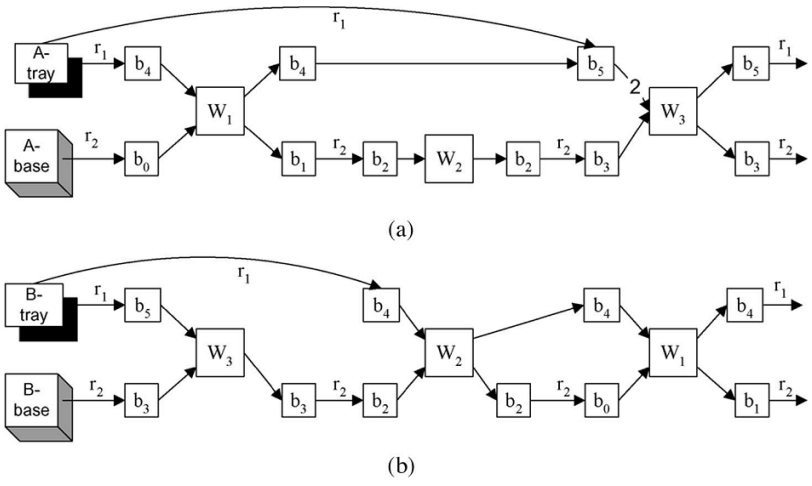


Figura 16: Processo de montagem de dois produtos concorrentes, proposto por Wu (2008)

Base de fluxo: assumindo que a capacidade dos buffers b_{0-3} é um e que a capacidade de b_{4-5} é mais do que um. Há uma base em b_0 e b_1 e duas peças da bandeja-A b_4, e ao mesmo tempo, existe uma base B em b_3. Agora, a uma base em b_1 ou a base B em b_3, b_2 pode mover-se em acordo com a Figura 16. Montagem: presumir que os dois buffers b_0 e B_4 têm uma capacidade de um lado, e b_0 e b_4 mantêm uma base de B e uma bandeja-A, respectivamente. Nenhuma operação pode ocorrer em w_1, eventualmente, se não é imediatamente, levando a um bloqueio do sistema. Finalmente, o fluxo do canal parte para resultar em um *deadlock*.

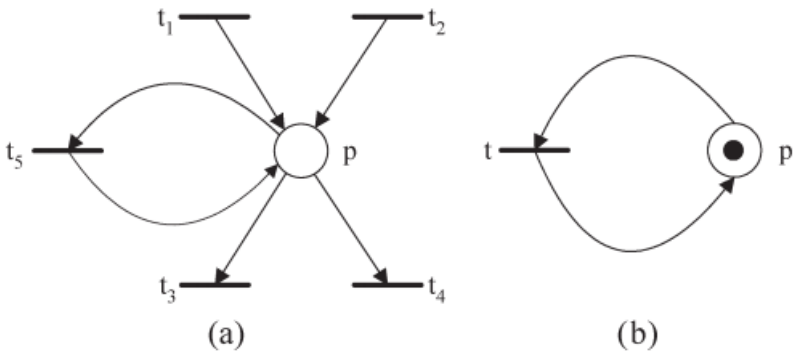


Figura 17: Modelo de rede de Petri para (a) Recurso -H e (b) Recurso G

Para evitar *deadlock* no FAS, o mecanismo de alocação dinâmica de recursos deve ser modelado. O FAS é modelado em Roszkowska (2004) por uma PN orientada a processos, onde um *place* de operação é introduzido para cada operação. Neste documento a proposta de Wu (2008) servirá de modelo usando um ROPN. Usando um modelo ROPN, cada recurso é modelado com apenas um único *place*, mas sem introduzir *places* de operação. Todos os roteiros de montagem são modelados através dos fluxos das marcações.

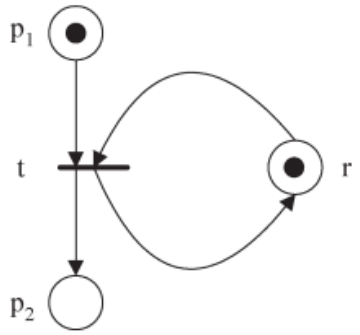


Figura 18: Modelo de rede de Petri para movimentação de peças por robô proposto por Wu (2008)

Se adequadamente modelados, alguns recursos irão contribuir para bloqueio do sistema, mas outros não. Assim, na modelagem do sistema, diferentes recursos devem ser tratados de maneiras diferentes. Portanto, Wu, Zhou e Li (2008) trata de forma diferente buffers de estações de trabalho e de robôs. O primeiro é chamado de recurso-H, e este último é chamado de recurso-G, como é mostrado na Figura 17.

Transições de múltiplas entradas (por exemplo, T_{1-2}) na Figura 18 para o modelo *p* significam que o recurso é requerido por vários processos. Transições de saída múltipla (por exemplo, T_{3-4}) significam as escolhas para a próxima operação.

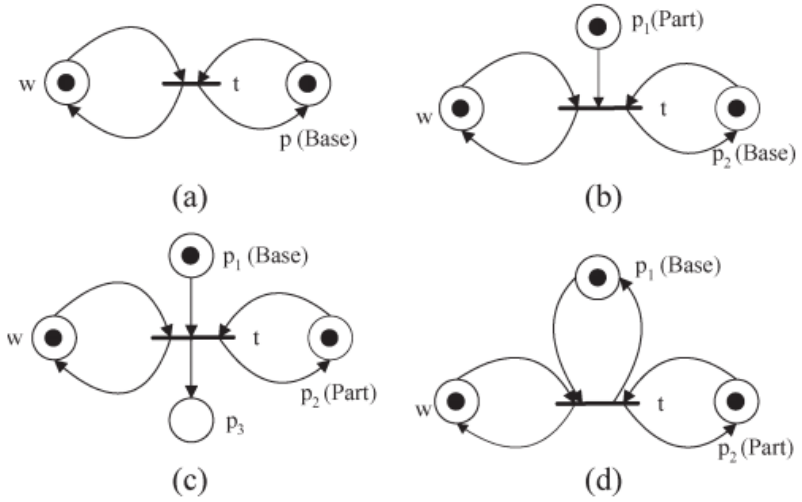


Figura 19: redes de Petri primárias por estação de trabalho. Propostas por Wu (2008)

Há quatro situações para as operações realizadas por uma estação de trabalho, como mostrado na Figura 19. (a) Uma estação de processamento de um componente de base em um *buffer* sem a utilização de qualquer peça. Após isso, a base permanece no *buffe*r. (b) Um componente de base está em um *buffer*. A estação usa uma peça de outra reserva para executar a operação. Após isso, a base permanece em seu *buffer*, mas a reserva, que mantém a peça é esvaziada. (c) Um componente de base está em um *buffer*. Uma estação de trabalho utiliza uma parte no *buffer* 2 com bandeja para executar a operação. Após a conclusão, a base é colocada do *buffer* 3 para o *buffer* 1 que está livre, mas o *buffer* 2 ainda está ocupado pela bandeja. (d) Uma estação de trabalho utiliza uma peça no *buffer* 2 para executar a operação no componente de base em um *buffer*. Após isso, a base e a bandeja permanecem nos *buffers* 1 e 2, respectivamente.

Com as primitivas de recursos e operações básicas, Wu, Zhou e Li (2008) apresenta o modelo de ROPN para os produtos individualmente, como é apresentado na Figura 20 para o produto A e Figura 21 para o produto B.

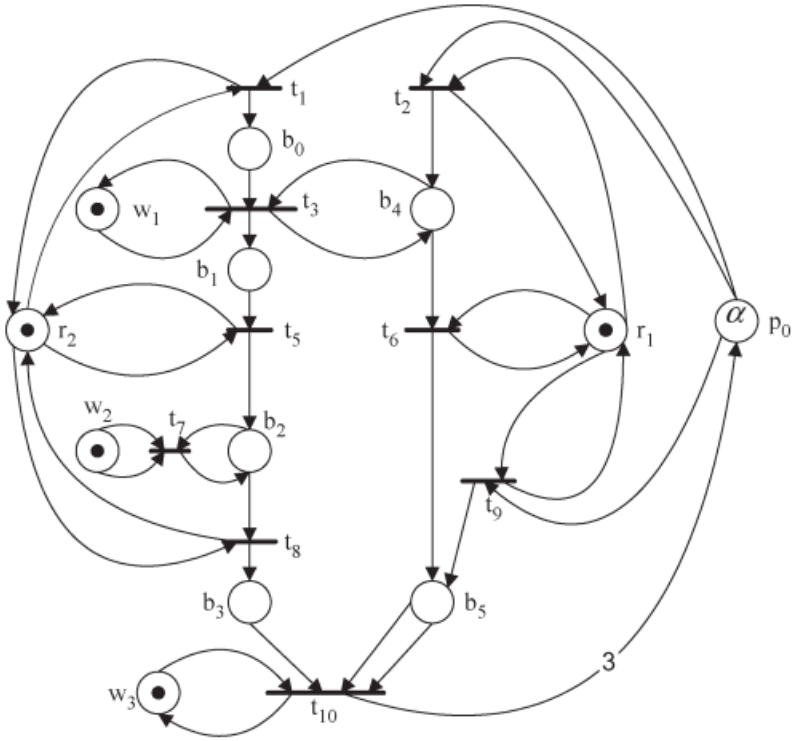


Figura 20: Sub rede para montagem do produto-A

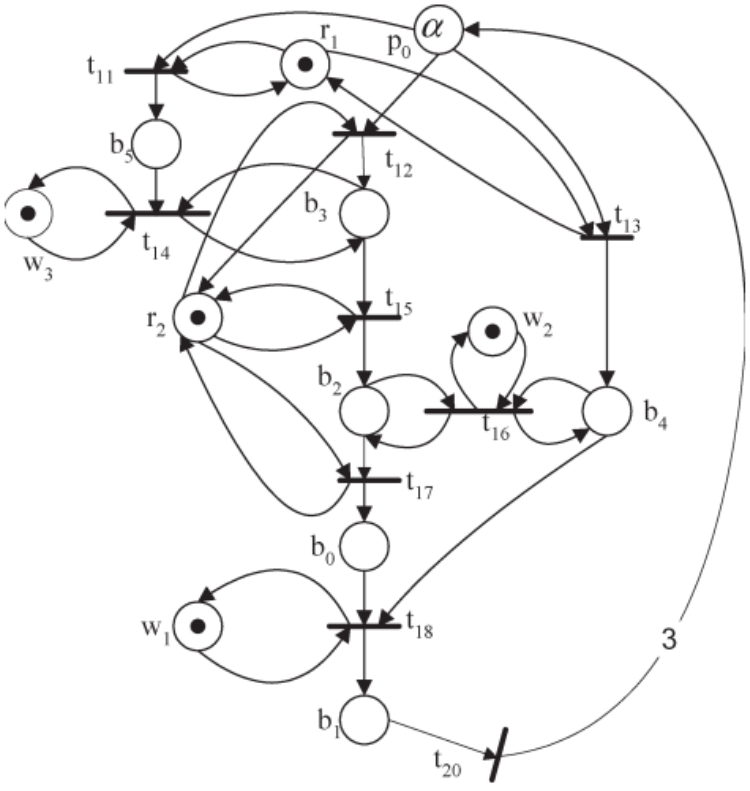


Figura 21: Sub rede para montagem do produto-B

Wu (1997) mostra que, através da modelagem dos recursos-G, os *deadlocks* resultantes de processos realizados por recursos-G podem ser eliminados. Assim, no sentido de evitar *deadlock*, os *places*-G e os seus arcos associados podem ser removidos do modelo. Ao removê-los, são reduzidos.

Depois de reduzidos, os modelos individuais são fundidos e o resultado é a ROPN apresentada na Figura 20.

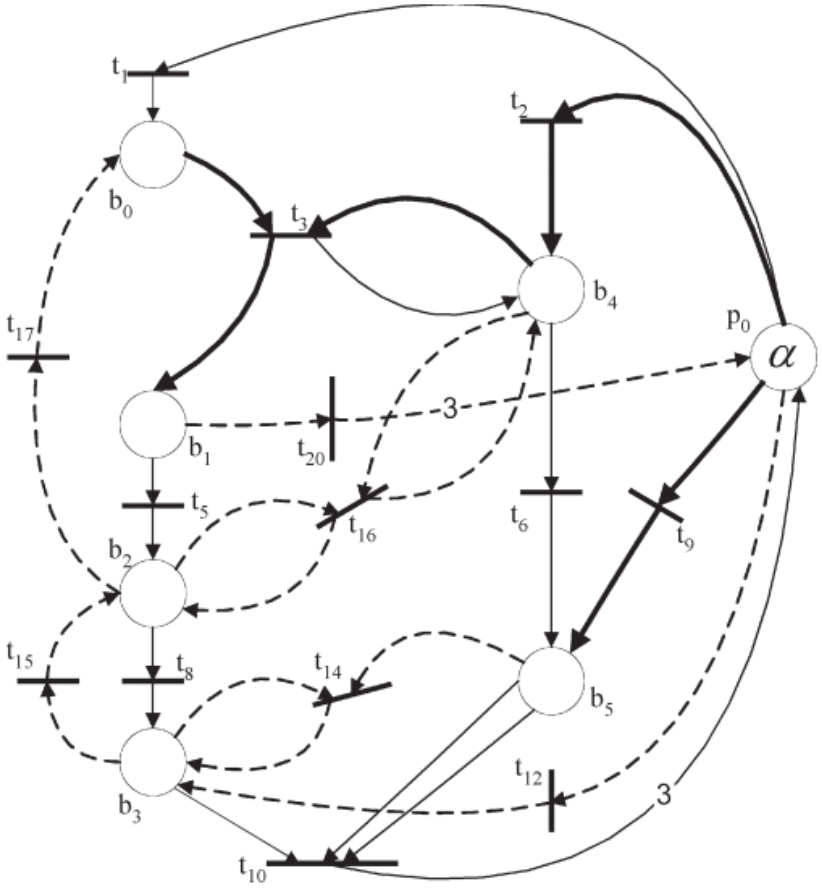


Figura 22: Modelo de rede de Petri orientada a recursos resultante

Descrição da técnica.

No processo de produção, cada peça é processada independentemente. Uma estação de trabalho permite somente uma peça no sistema e a peça processada pode ser concluída uma por vez. Nesse meio, o processo de produção sempre será realizável (p6).

Observa-se que ROPN apresentado para processos de montagem. A base do fluxo de componentes, em algum sentido, é similar à parte do processo de produção. Para o fluxo da base na Figura 22, existem dois circuitos C1={b0,t3,b1,t5,b2,t17b0} e

C2={b2,t8,b3,t15,b2}. Podem ocorrer *deadlocks* nesses circuitos assim como os *deadlocks* em parte do processo de produção.

WZL-Policy: Uma transição $t \in Y$ na ROPN para FAS na marcação M é habilitada e esta dispara mudanças de M para M'. Então, t pode disparar somente se as seguintes condições são satisfeitas:

1. Assume se que existem k grupos de *buffers* B_{1-k} para a base de componentes. Então, em M', existem, pelo menos, $k-1$ espaços livres de *buffer* tais que, no máximo, um grupo B_i está cheio.

2. Assume se que existem k *buffers* B_{1-k} para a peça a ser montada em cima dos componentes base. Então, em M', existem, pelo menos, $k-1$ espaços de *buffers* livres tais que, no máximo, um *buffer*s está cheio.

3. Assume se a transição $t \notin T_{Tray}$, e seu disparo move marcações V_h dos produtos tipo-h em p_i. Logo: $M(p_i)\,(k)$ significa o número de marcações representando produtos do tipo k em $p_i$na marcação M. Suponha que $t_1 \in T_{Tray}$, e seu disparo move a marcação U_h de produtos do tipo h em p_i. Então,

   a. $\mathrm{K}(p_i) - W[i] - \Sigma_{k \in AS(i), k \neq h} \max(M(p_i)(k), R_k[i]) - M(p_i)(h) - U_h \geq V_h, se\ h \in AS(i)$;

   b. $\mathrm{K}(p_i) - Z[i] - M(p_i)(h) \geq V_h, se\ h \in NAS(i)$.

4. Assume se que $t \in T_{Tray}$, seu disparo move marcações U_h de produtos tipo-h com cor C_1 em p_i, essas marcações juntas com marcações V_h de produtos do tipo h em p_j e marcações Y_h com cores C_2 em p_i, habilita a transição de montagem t_a. Então $M(p_j)(h) \geq V_h, M(p_i)(C_2) \geq Y_h$ e $M(p_1)(C_1) = 0$.

As condições 1 e 2 garantem que o componente base e o fluxo da peça não serão bloqueados respectivamente. A condição 3 garante que o estado apresentado na Figura 23 nunca ocorrerá. A condição 4 evita qualquer disparo antecipado de modo que o espaço do buffer esteja ocupado. Observa se que a condição 3 não faz restrições em *places* que não são da montagem. Um *place* que não é da montagem necessita adequar-se somente as condições 1 e 2.

Norma 1: Uma sub rede CROPN inicialmente marcada com n *places* é dita livre se, em qualquer marcação M, existem, pelo menos, $n-1$ *places* livres habilitados tais que, no máximo, um *place* está cheio.

Prova: É preciso mostrar somente que, se a sub rede tem somente $n-1$ espaços livres e no mesmo tempo a distribuição das condições é satisfatória, a subrede é livre de *deadlock*. Isto é porque, se existem mais espaços livres, o problema é mais simples.

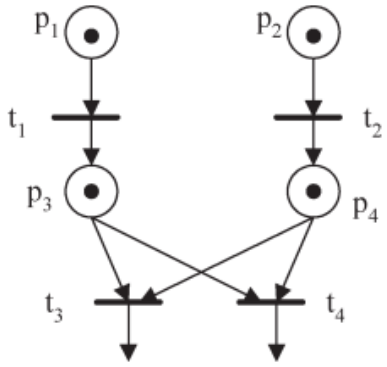


Figura 23: Subrede para operações de montagem por Wu (2008)

Resultados

Segundo Wu, Zhou e Li (2008), a política de controle de *deadlock avoidance* proposta mostrou-se computacionalmente eficiente e melhor do que a proposta de Hsieh (2004) a qual foi comparada.

Em Hsieh (2008), uma junção sistema de montagem foi estudada, onde os recursos são máquinas. A máquina realiza operações e detém as peças ao mesmo tempo. Tal sistema pode ser modelado pelo método apresentado, com algumas modificações. A política de controle pode ser aplicada também. Um estudo de caso industrial foi usado para mostrar os resultados.

3.4.3 Outras Abordagens

Além do uso de redes de Petri nas estratégias de resolução de *deadlock* em SMA na alocação de recursos, também são encontradas na literatura estratégias alternativas, os mais comuns encontrado são, o uso de dígrafos, algoritmo do banqueiro, controladores robustos e planos de processos parcialmente ordenados. A seguir serão apresentadas algumas dessas técnicas.

Teoria dos grafos

Segundo Fanti (2002) dígrafos são ferramentas mais sintéticas em relação ao uso de redes de Petri, pois eles representam somente os recursos do sistema, como em Maione e DiCesare (1998). O uso de dígrafos para estratégia de resolução de *deadlock* em SMA descreve as interações entre as tarefas e os recursos do processo de produção para a caracterização do *deadlock* de forma concisa.

Em Fanti (2002) foram comparados a técnica que utiliza dígrafos para *deadlock avoidance* e a que utiliza redes de Petri. O resultado da proposta mostrou como a política de *deadlock avoidance* resultante da matriz de incidência gerada pela teoria dos grafos pode ser implementada com mais facilidade por redes de Petri.

Algoritmo do banqueiro

A abordagem do algoritmo do banqueiro, como as outras abordagens voltadas para a resolução de *deadlock avoidance*, baseia-se em procedimentos de decisão usando informações do sistema para manter um sistema de controle livre de *deadlock*.

O algoritmo do banqueiro, proposto por Dijksta (1965) utiliza informações sobre quais recursos um processo requer e quantos recursos estão disponíveis no sistema, assim o algoritmo pode liberar o processo ou bloqueá-lo até que o mínimo de recursos esteja disponível no sistema. Este algoritmo, inicialmente, foi proposto por Dijksta (1965) para evitar *deadlock* em sistemas computacionais.

Ezpeleta (2002) propôs o uso do algoritmo do banqueiro para evitar *deadlock* em FMSs do tipo SU-RAS, utilizando informações do sistema, assim como, estações de produção e elementos do conjunto de movimentação e armazenagem de peças, para calcular os recursos disponíveis e quantos recursos um procedimento de produção irá utilizar. Assim que os cálculos são feitos o controlador permite ou inibe o processo de produção requisitado.

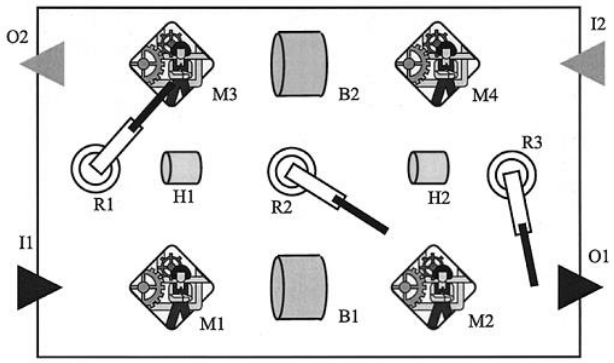


Figura 24: Célula de Manufatura Flexível usado por Ezpeleta (2002)

O custo para aplicar o algoritmo em um sistema relativamente pequeno (quatro estações de processamento e três robôs de manuseio), como é apresentado na Figura 24, mostrou-se um problema polinomial. Porém o algoritmo não se mostrou maximamente permissivo, ou seja, os recursos forma utilizados de forma conservadora impedindo um uso mais eficiente do sistema.

3.5 Considerações Finais

Neste capítulo foram estudadas várias abordagens voltadas para a resolução de *deadlock* em SMA, cada estratégia pode trabalhar em níveis diferenciados do SMA (design, planejamento, programação e controle). Porém, o foco da pesquisa foi na resolução do problema de *deadlock* nos estágios de programação e controle.

As abordagens que usam estratégias de resolução *Deadlock Prevention* usando modelos de rede de Petri encontram um desafio em relação a complexidade da rede gerada, ou seja, quanto mais complexo o SMA, maior será o modelo gerado.

As abordagens que usam modelos redes de Petri aplicando políticas de controle *Deadlock Avoidance*, se deparam com o desafio da permissividade. É apresentada na Figura 25 a relação de permissividade entre as técnicas utilizadas. Quando a política de controle se mostra restritiva, significa que a tendência é ter menos recursos sendo utilizados simultaneamente no SMA, apesar do custo computacional e a complexidade do problema tender a ser polinomial, por outro lado, a tendência do desempenho do SMA é cair. Agora, utilizando uma política de controle *Deadlock Avoidance* mais permissiva, o desempenho do SMA tende a aumentar, porém isso pode aumentar também a complexidade do problema e o custo computacional.

Figura 25: Relação de permissividade entre as políticas de controle *Deadlock Avoidance* em SMA

Em conformidade com os tipos de modelagens estudados, os modelos de rede de Petri possuem atributos que podem representar grande parte das características de um

sistema de manufatura automatizado. A abordagem modular, encontrada nas redes de Petri Coloridas, entre outras, tem como objetivo tratar partes diferentes do sistema de manufatura, facilitando modelagem e representação do sistema. Dentre as técnicas abordadas, pode-se concluir que, apesar de que a utilização de dígrafos, e outros algoritmos, terem vantagens específicas, as redes de Petri são modelos sistemáticos, modulares para análises, simulação, planejamento e controle de FMSs (FANTI et. al, 2002).

A partir dos estudos e pesquisas realizados nos capítulos anteriores, o Capítulo 4 apresenta a proposta deste trabalho como segue.

Capítulo 4

PROPOSTA

4.1 Considerações Iniciais

Nos capítulos anteriores foi abordado o problema de *deadlock* no controle de Sistemas de Manufatura Automatizados, com isso, algumas abordagens, que buscam formas de solucionar o problema sem que isso comprometa o desempenho desse tipo de sistema, foram levantadas, tornando-o assim, mais confiável.

Em conformidade com os tipos de modelagens estudados, os modelos de rede de Petri possuem atributos que podem representar grande parte das características de um sistema de manufatura automatizado. A abordagem modular, encontrada nas redes de Petri Coloridas, entre outras, tem como objetivo tratar partes diferentes do sistema de manufatura, facilitando modelagem e representação do sistema. Dentre as técnicas abordadas, pode-se concluir que, apesar de que a utilização de dígrafos, e outros algoritmos, terem vantagens específicas, as redes de Petri são modelos sistemáticos, modulares para análises, simulação, planejamento e controle de FMSs (FANTI et. al, 2002).

A partir dos estudos e pesquisas realizados nos capítulos anteriores, Neste capítulo será apresentada a proposta deste trabalho como segue.

4.2 Restrições abordadas

Considerando o escopo e objetivos deste trabalho, algumas restrições serão colocadas para melhor compreensão dos resultados obtidos. Um SMA real possui atributos e fatores que exigem um alto grau de complexidade para serem considerados, se esses

fatores forem considerados, os eventos no sistema não seriam determinísticos ou pertencentes a um conjunto finito de estados, fazendo com que a modelagem e aplicação da estratégia tornem-se desvantajoso ou até inviável. As restrições consideradas são:

- Em cada passo do processo de produção uma peça requer uma simples operação de um único tipo de recurso, ou seja, considera-se o SMA como um Sistema de Alocação de Recursos de Unidade Simples (SU-RAS).
- Desgastes naturais de ferramentas das estações de produção não são considerados;
- Máquinas nunca falham (Estações automatizadas, robôs, Veículos auto-guiados e outros);
- A comunicação entre o controlador e o SMA nunca falha;
- Os tempos de operação, setup, transporte e movimentação são previamente conhecidos;
- Uma vez que uma determinada operação é iniciada, esta não é interrompida até que seja concluída.

É importante observar que, mesmo que essas restrições são levadas em conta. É possível a partir do método de modelagem proposto, simular falhas de comunicação ou qualquer tipo de situação que simbolizam a ausência de um ou mais recursos. Também é possível desenvolver modelos com tempos de operação aleatórios dentro de um limite mínimo e máximo utilizando a ferramenta de modelagem (*CPNTools*). Com isso é possível estudar comportamento do sistema com falhas e realocação dos recursos.

4.3 Estrutura do controle do sistema

Assim que uma nova produção entra no sistema de manufatura, primeiramente esta entra na programação da produção (ver Figura 26) a partir de informações atualizadas dos estados do processo de produção em andamento.

Com essas informações é criada no sistema uma ordem de produção, onde informações como, tipos de operação e prioridades de regras são passadas para o controle de produção que contém informações do dos estados do sistema de manufatura e também a base de regras e regras de prioridade dos produtos. Com essas informações recebidas em tempo de operação, o controlador de *deadlock* envia comandos de permissão ou inibição de eventos para o sistema de manufatura.

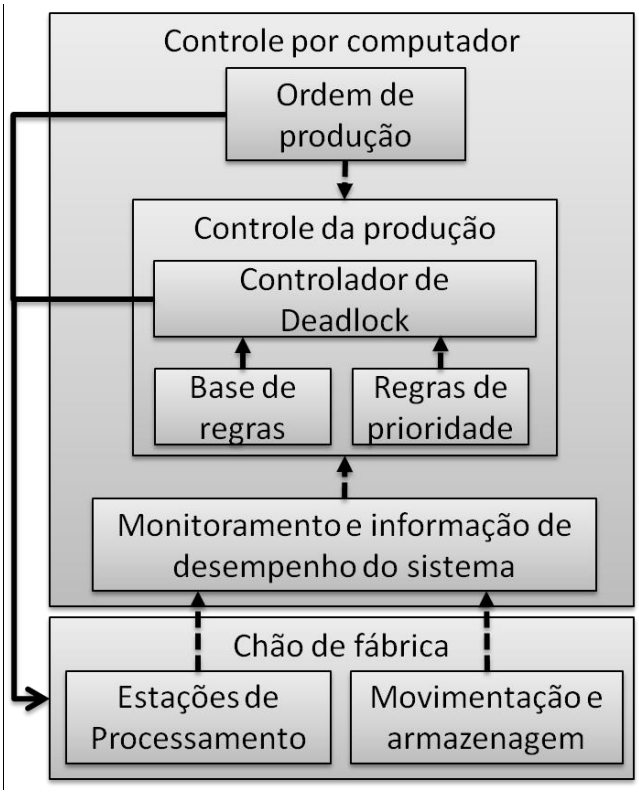


Figura 26: Estrutura do controle

4.4 Método de modelagem

Os cenários foram modelados em rede de Petri Colorida Temporizada usando do *software* de modelagem CPNTools. Deste modo, permitindo que as análises necessárias das propriedades da rede de Petri sejam feitas.

O padrão de representação da rede de Petri Colorida é dado pelo *software* de modelagem CPNTools (ver Figura 27) onde, os *places* são representados por elipses, as transições são representadas por retângulos, os arcos são representados por setas e as marcações são descritas na parte superior direita dos *places*. O conjunto de cores pertencente ao *place* é descrito na parte inferior da elipse que representa o *place* em questão.

Os arcos são subdivididos entre arcos de entrada (os que são representados por setas que apontam para um *place*) e arcos de saída (representados por setas que apontam para uma transição). Os arcos de saída determinam uma condição para que a marcação presente no *place* seja consumida pela transição. Os arcos de entrada determinam quais marcações serão geradas pela transição dependendo ou não dos arcos de saída.

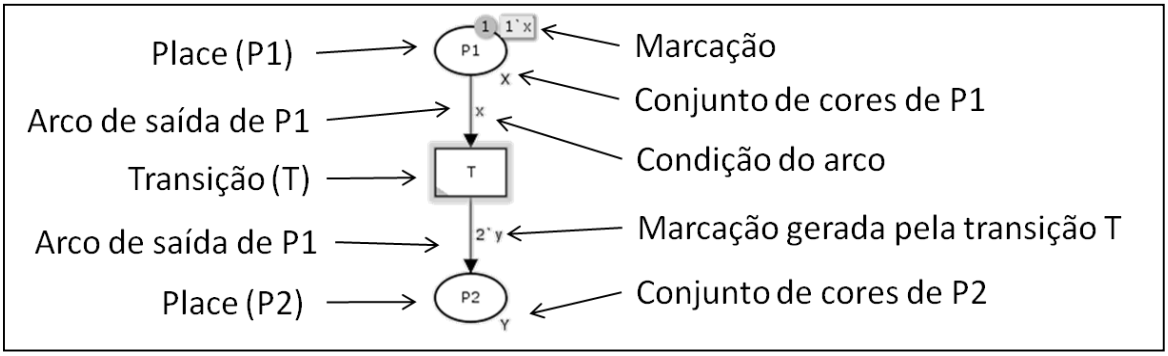


Figura 27: Representação gráfica de uma rede de Petri pelo CPNTools

As regras de modelagem descritas a seguir são importantes para a descrição da técnica de resolução de *deadlock* proposta que será descrita no Item 0, pois, a mesma requer uma modelagem sem que haja conflitos e ambiguidades na rede. A seguir serão utilizados alguns exemplos de sistemas e modelos menos complexos para que as regras sejam esclarecidas de forma direta e didática. O método de modelagem é baseado nos recursos existentes no sistema e roteiros de produção de possíveis tipos de produtos ou peças que o SMA pode produzir.

4.4.1 Ordem de produção

Quando uma ordem de produção entra no sistema de controle, durante o processo de produção, o controlador recebe esta informação, a informação recebida é interpretada para o modelo. Assim que a informação é interpretada, caso o controlador autorize, dispara-se uma transição que gera uma marcação referente à nova peça que entrou no processo de produção.

4.4.2 Operação

Neste trabalho será considerado como operação qualquer procedimento de montagem ou usinagem que uma estação de trabalho execute em uma peça ou produto durante o processo de produção. Na Figura 28 é apresentada uma operação de montagem de um FMS.

Assim que uma peça chega ao *buffer* representado pelo *place B1* uma marcação *p* pertencente a cor *PRODUTO* também estará em *B1*. Quando a estação de produção que irá fazer a montagem da peça está disponível, de modo paralelo, uma marcação *r* do conjunto de cores *RECURSO* estará no *place M1*. Assim que as duas marcações estão disponíveis, a transição *T1* é habilitada e pode ser disparada. Quando a transição é disparada, esta consome as marcações *p* e *r* e geram marcações *p* e *r* novamente no final do disparo.

Enquanto a transição é disparada a estação de trabalho *M1* estará indisponível para executar outra tarefa, já que o processo não é preemptivo.

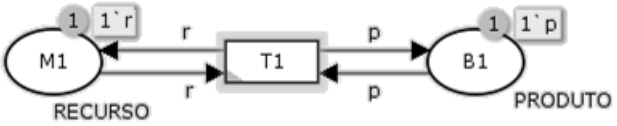


Figura 28: Operação de montagem

4.4.3 Movimentação e Transporte

Na Figura 29 é apresentado como é modelado o processo de movimentação de peças, onde o *place R1* representa um robô de movimentação. A transição *T12* representa o processo de movimentação, ou seja, quando uma peça se encontra no *buffer* representado pelo *place B1* e o robô *R1* está disponível, então a transição *T12* é habilitada e pode ser disparada. Quando a transição é disparada ela consome a marcação *p* da cor *PPRODUTO* e *r* da cor *RECURSO* então são geradas marcações *r* e *p* para os *places R1* e *B2* respectivamente.

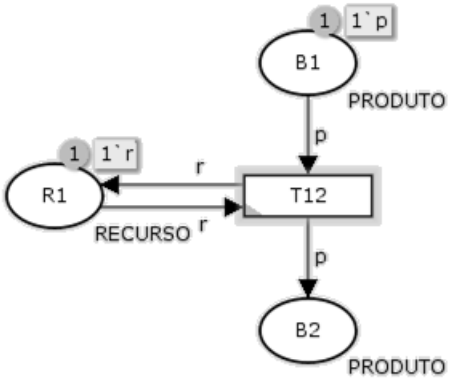


Figura 29: Processo de transporte

4.4.4 Etapas de produção

A Figura 30 apresenta um exemplo simples para melhor compreensão das etapas de produção que serão mostradas. Na figura existem dois *buffers* (*B1* e *B2*), sendo que o *buffer B1* pertence à Estação de produção (M1) e um robô de movimentação (*R1*). Neste exemplo um produto *p* encontra-se inicialmente no *buffer* B1, as etapas de produção do produto é ser usinado pela estação de trabalho *M1* e depois ser transportada do *buffer B1* para o *buffer B2* pelo robô *R1*.

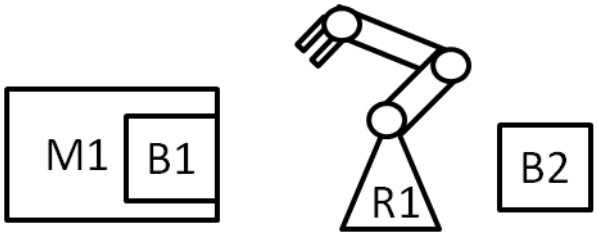


Figura 30: Exemplo para roteiro de produção

De acordo com as definições de operação e movimentação propostas nos itens anteriores, na Figura 31 é apresentada a junção dos dois eventos descritos em sequência, conforme as definições. Porém, somente a utilização das definições deixa a rede de Petri resultante não determinística. Exemplificando, a transição *T1* representa a operação de montagem do produto *p* pela estação de produção *M1*, essa transição só é habilitada se a marcação *p* está em *B1* e a marcação r está em *M1*, ao mesmo tempo, a transição *T12*, que representa o processo de movimentação do produto *p* do *buffer B1* para o *buffer B2* pelo robô *R1*, só é habilitada se existir uma marcação *p* no *place B1* e uma marcação *r* no *place R1*. Portanto, as transições *T1* e *T12* estão habilitadas ao mesmo tempo, além do conflito gerado entre as transições, a transição *T12* pode ser disparada antes da *T1*, anulando as condições do roteiro de produção do produto *p*.

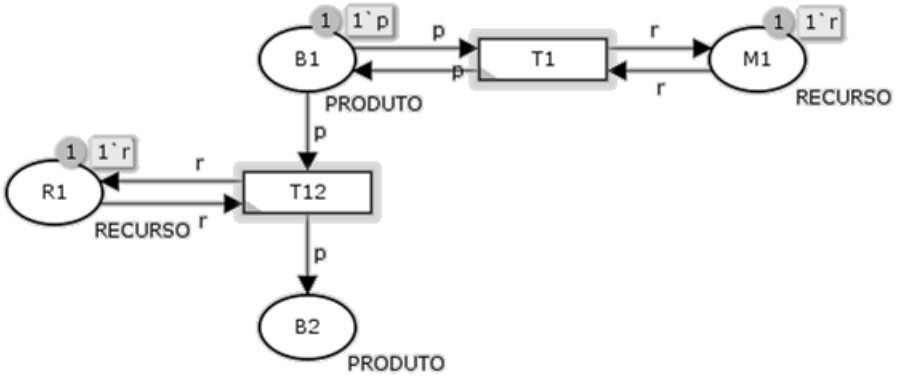


Figura 31: Conflito entre etapas de produção

Em razão às considerações anteriores, é proposto que, a marcação colorida de cor *PRODUTO* seja definida como segue:

$PRODUTO = \{p_i(e), \dots, p_n(e)\}$ onde $e \in E = \{1, \dots, k\}$.

A cor definida como *E* representa as etapas do processo de produção de um determinado produto. No exemplo que está sendo abordado, pode se considerar que o produto *p* possui duas etapas (*k=2*) conforme o roteiro de produção. Na primeira etapa o produto está localizado no *buffer B1* o que indica que ele ainda não passou pela operação da estação de trabalho, depois que o produto sai da operação ele é transportado para o próximo *buffer*, iniciando assim, a segunda etapa de produção.

Na Figura 32 é mostrada uma rede de Petri onde o produto *p*(1) está no *place B1* com o arco de saída $a(B1, T1, p(1))$, ou seja, a transição *T1* só é habilitada se no *place B1* existe uma marcação $p(e)$ onde $e = 1$, logo, no estado em que a rede de Petri se encontra a transição *T1* está habilitada e pode ser disparada, agora, a transição *T12* está desabilitada, porque o arco de saída $a(B1, T12, p(2))$ só permite marcações $p(e)$onde $e = 2$.

Depois que a transição *T1* é disparada, esta gera uma marcação $p(2)$ no *place B1*, isto indica que o produto já passou pela operação de *M1*, então, a transição *T12* é habilitada e pode ser disparada fazendo com que o roteiro de produção de *p* seja respeitada.

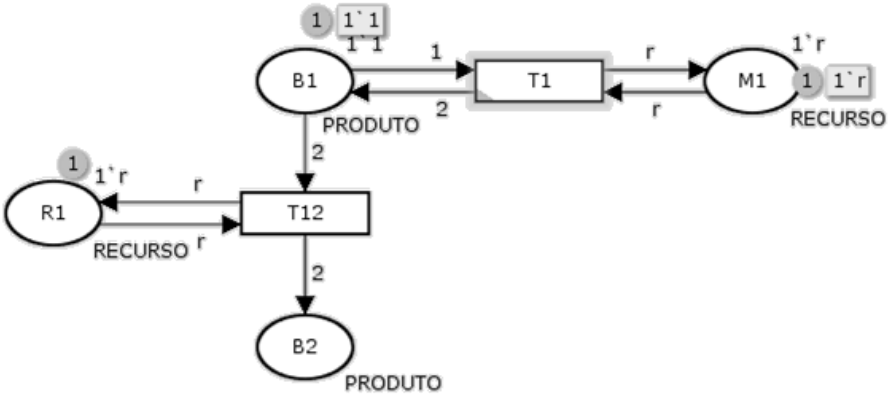


Figura 32: Etapas de produção sem conflitos

4.4.5 Fusão de roteiros de produção

Como visto nos capítulos anteriores, uma das principais características de um SMA é o compartilhamento de recursos, também foi visto que FMSs podem produzir tipos diferentes de produtos simultaneamente, portanto, no cenário ilustrado na Figura 33 tem-se parte de um FMS onde um robô (R1) está programado para mover peças entre os *buffers* *B1*, *B2* e *B3* de suas respectivas estações de produção *M1*, *M2* e *M3*.

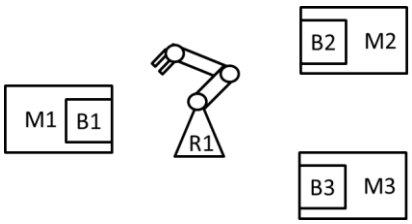


Figura 33: Cenário com três estações de produção e um robô de movimentação

A partir do *layout* definido, na Figura 34 são definidos dois roteiros de produção para os produtos p_1 e p_2. O produto p_1 passa pelo processo de operação da estação de trabalho *M1* então ele é transportado pelo robô de movimentação *R1* até a estação de trabalho *M2*.

O produto p_2 depois que termina sua operação em *M1* é transportado para a estação *M3* por *R1.*

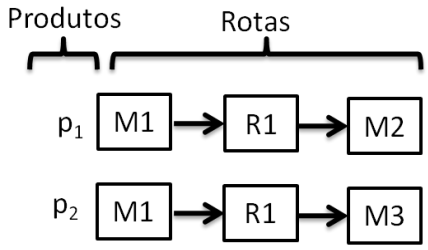


Figura 34: Roteiros de produção do cenário de exemplo

A partir dos dados obtidos do cenário de exemplos das definições de operação e transporte do método de modelagem, tem-se o modelo de CPN do roteiro de produção do produto p_1 (Figura 35) e do produto p_2 (Figura 36).

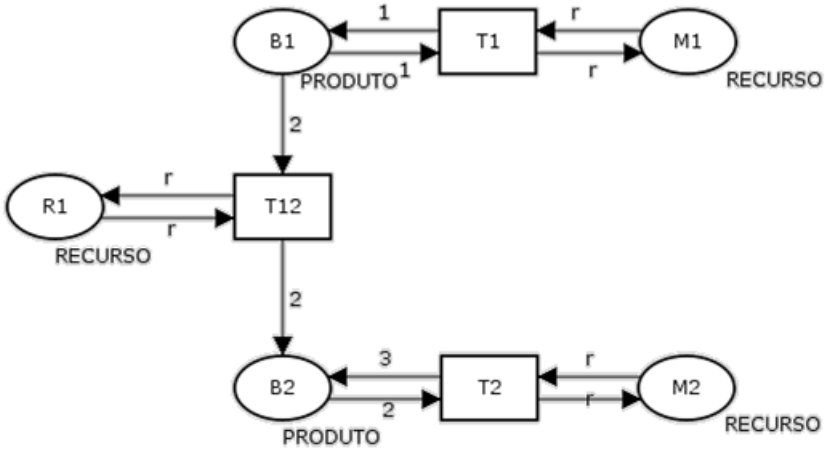


Figura 35; Roteiro de produção do produto p_1

Se os roteiros dos produtos fossem controlados separadamente, as informações poderiam entrar em divergência quando as peças requisitassem o mesmo recurso, portanto, um *deadlock* seria inevitável. Nota-se que os recursos *B1*, *M1* e *R1* são requisitados tanto pelo produto p_1 quanto pelo produto p_2. Mesclando os dois modelos resultantes, em outras palavras, adicionando *places* e transições complementares sem duplicar os *places* correspondentes ao mesmo recurso, obtém-se o modelo mostrado na Figura 37.

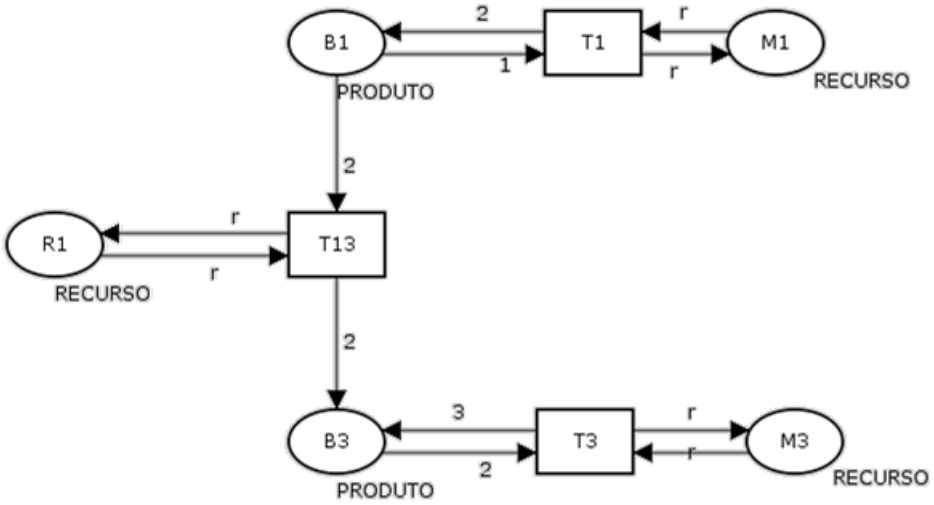


Figura 36: Roteiro de produção do produto p_2

As transições seguem um padrão de nomenclatura, assim, a compreensão de qual evento do SMA que cada uma representa se torna clara. As transições de operação recebem os números correspondentes à estação de produção que é controlada e o tipo de produto (Ttr onde t corresponde ao tipo do produto que será usinado e r representa a estação de produção que o comando irá enviar comandos e receber informações). As transições de transporte são nomeadas com informações do tipo de produto e dos *buffers* de origem e destino da peça que será deslocada ($Ttod$ onde t representa o tipo de produto da peça que será transportada, o corresponde ao *buffer* de origem e d refere-se ao *buffer* de destino da peça).

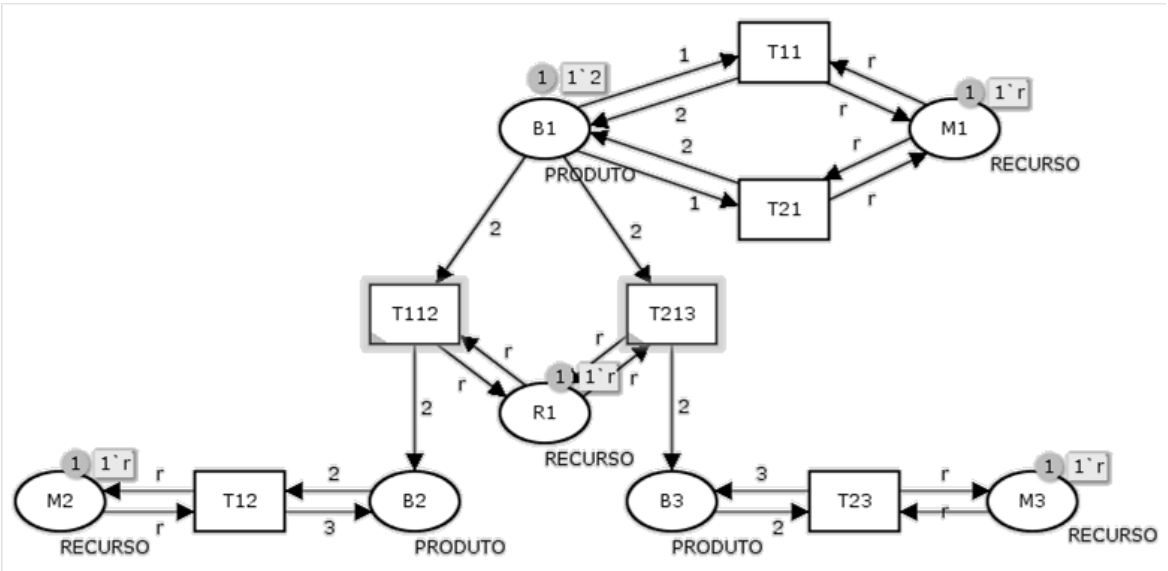


Figura 37: Fusão dos roteiros de produção dos produtos p_1 e p_2

Na Figura 37 é apresentado um estado do modelo de CPN onde o *place B1* possui uma marcação da qual não se sabe o tipo de produto que se refere. Logo, existe um conflito entre as transições *T112* e *T213*. Com a adição de uma nova informação na marcação, é possível resolver este conflito. Assim, o conjunto de cores de marcações, que representa uma peça de um produto em processo de produção, recebe uma informação adicional. Logo, a marcação de cor *PRODUTO* é definida como segue:

$PRODUTO = \{p_i(t,e), \dots, p_n(t,e)\}$ onde $t \epsilon TIPO = \{0, \dots, n\}$ e $e \in E = \{1, \dots, k\}$. n é u número de tipos de produtos que podem ser processados no sistema de manufatura e k é o número de etapas que as peças podem atingir no decorrer do processo de fabricação.

Com essa nova informação na marcação, o problema apresentado é resolvido. A Figura 38 apresenta a mesma situação da Figura 37, porém sem conflito, ou seja, a marcação possui a informação do tipo de produto que a peça representa, com a condição nos arcos de acordo com o tipo de produto, somente a transição *T112* foi habilitada e pode ser disparada enviando o comando para o robô *R1* para transportar o produto p_1 do *buffer B1* para o *buffer B2*.

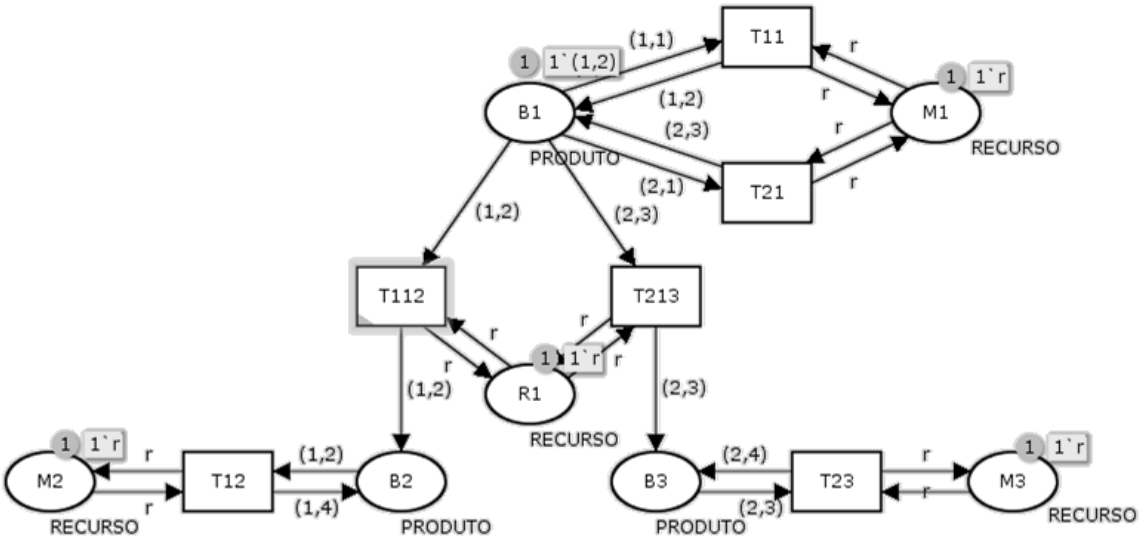


Figura 38: Rotas de produção dos produtos p1 e p2 sem conflito

4.4.6 Limitações

Em SMA normalmente, os *buffers* de armazenamento existentes no decorrer do processo de produção possuem capacidades limitadas de armazenamento. O exemplo apresentado na Figura 39 ilustra uma rede de Petri onde existem duas marcações, sendo que, uma está no *place B1* e outra no *place B2*. As transições *T12* e *T23* estão habilitadas

para o disparo, porém, se a transição *T12* disparar antes da transição *T23*, o *buffer* estará com duas marcações ao final do disparo. Esta situação pode ser um problema se no SMA que a rede representa o *buffer B2* tem capacidade para armazenar somente uma peça por vez. Mesmo que o *buffer* tenha capacidade para duas peças por vez se na rede tivesse uma terceira marcação em *B1* a transição *T12* estaria habilitada dando permissão para o transporte de uma terceira peça para um *buffer* que não tem capacidade.

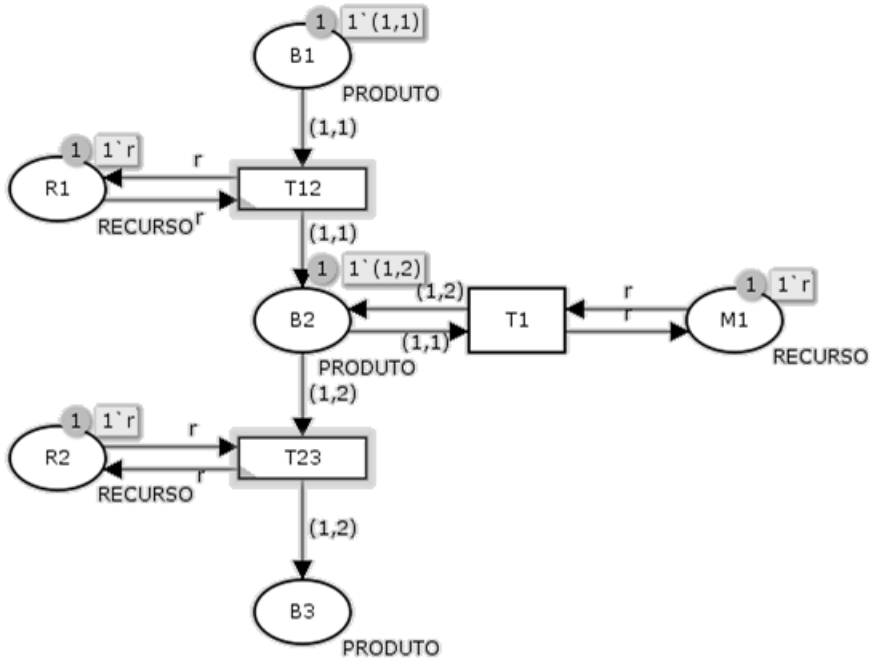


Figura 39: Buffers sem limitações de armazenamento

A resolução destes problemas na modelagem e no controle do SMA é proposta com a adição de “marcações vazias”, em outras palavras, são marcações que indicam quantos espaços vazios existem no *place*. Além disso, são adicionados arcos de marcações vazias na entrada e saída dos *places* com limitações.

Na Figura 40 o *place B3* possui uma marcação vazia. Quando a transição *T23* é disparada esta consome a marcação vazia, gera a marcação que representa a peça que foi transportada e, por fim, gera outra marcação vazia no *place B2*, liberando a transição *T12* para ser disparada que até então não estava habilitada, pois não existia uma marcação vazia em *B2* para ser consumida.

Caso um *buffer* tenha a capacidade de armazenamento de no máximo três peças por vez, por exemplo, então o *place* que o representa deve ter três marcações vazias no momento em que o *buffer* estiver vazio.

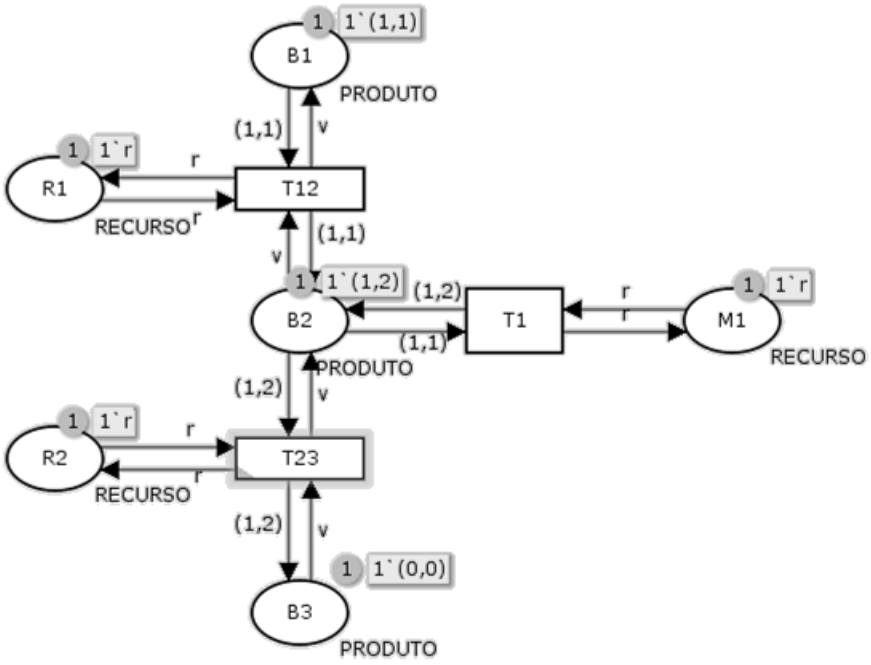


Figura 40: Limitação de buffers

4.4.7 Módulos

A modelagem com CPN permite usar a abordagem modular, ou seja, o modelo pode compor dois ou mais níveis, facilitando a visualização da rede e dos processos do sistema. A modelagem a seguir remete ao controlador de *deadlock* ilustrado na Figura 42 onde a leitura dos dados e o envio de comandos para o controle do sistema são tratados através de módulos, que enviam e recebem informações em tempo de operação.

Na Figura 41 é apresentado como funciona uma rede modular de dois níveis, na figura, a transição do módulo superior remete a rede do módulo inferior e os *places* ligados por arcos à transição na rede superior são as entradas e saídas para a rede do nível abaixo. Da mesma forma a rede apresentada na Figura 42 é um módulo usado nas transições de operação da Figura 28 e de movimentação da Figura 29.

Na Figura 42, a transição *OUT* envia comandos para que um robô execute uma tarefa de movimentação de peças entre as estações de produção do SMA. Assim que o robô termina a execução, a transição *IN* é disparada e por fim, a transição *OK* dispara, liberando o robô de movimentação para outras tarefas.

Os módulos também são usados para a execução dos processos de operação das estações de produção.

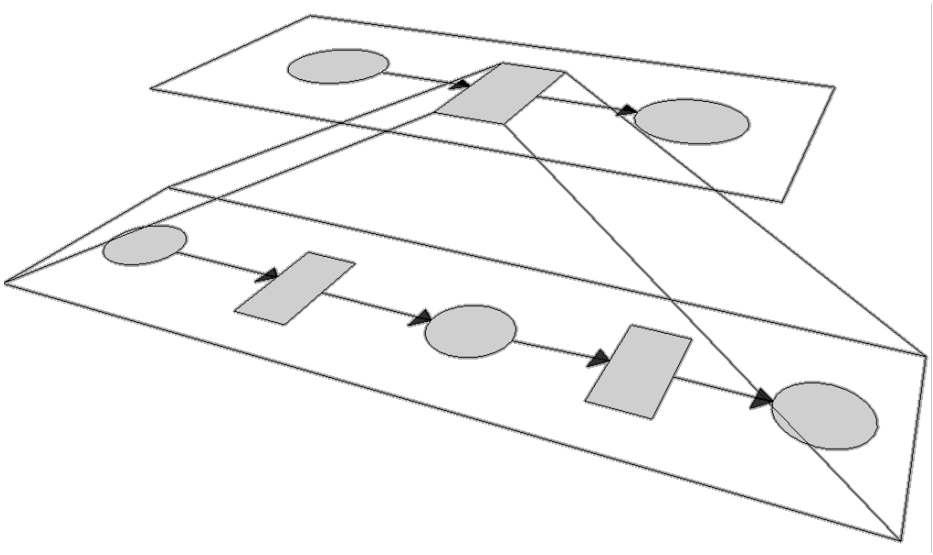


Figura 41: Rotas de produção dos produtos p1 e p2 sem conflito: Exemplo de módulo em uma rede de Petri

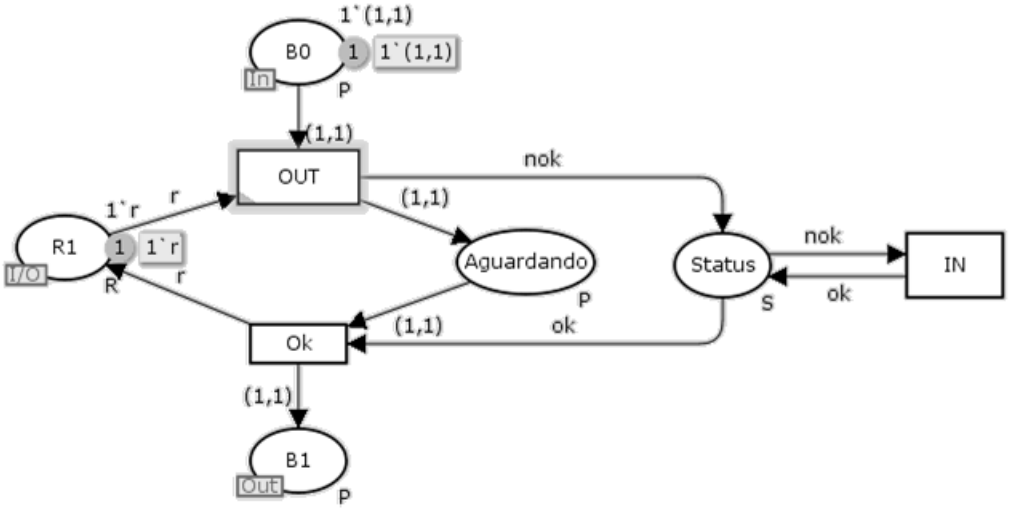


Figura 42: Módulo de entrada e saída do controlador

4.5 Técnica de Resolução de *Deadlock*

No item anterior, foi definido o método de modelagem proposto neste trabalho. Observa-se que o método é voltado à alocação de recursos em relação ao roteiro de produção previamente definida pelos cenários propostos. Para que a estratégia de resolução de *deadlock* seja aplicada a um SMA, primeiramente, deve-se levar em consideração o arranjo do SMA, todos os recursos disponíveis, a capacidade de cada *buffer*,

todas as operações possíveis de cada recurso e os roteiros de produção de todos os produtos que podem ser fabricados pelo sistema.

Com as informações e o método de modelagem proposto é obtido o modelo de CPN do SMA. A partir disso, é possível identificar regiões que dão condições para a ocorrência de *deadlock*. Depois que as regiões são identificadas, conforme mostrado no Item 3.1, a técnica é aplicada. A técnica se resume em impedir eventos que levam o sistema a um estado de *deadlock* ou a um estado que antecede o estado de *deadlock*, pois a partir desde, já não é possível impedir o próximo estado. Uma região crítica é um conjunto de recursos que são requisitados por tipos diferentes de produtos, em outras palavras, são regiões onde podem ocorrer *deadlocks*.

Na Figura 43 é apresentado um cenário no qual existe uma região crítica que pode ocorrer *deadlock*. No cenário existem dois produtos (p_1 e p_2) com seus roteiros de produção descritas na Figura 44. A partir do método de modelagem o modelo de CPN pode ser obtido, como é apresentado na Figura 45.

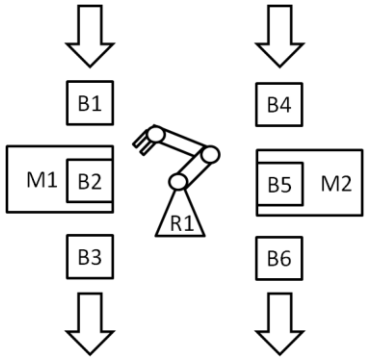


Figura 43: Cenário de exemplo para a técnica de DA proposta

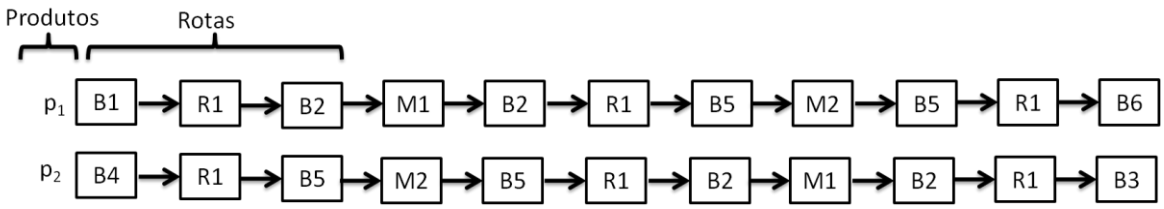


Figura 44: Rotas dos produtos p_1 e p_2

A partir do modelo obtido, o conjunto de transições é observado como segue:

$$T = \{T101, T204, T11, T21, T22, T12, T112, T125, T156, T245, T252, T223\}$$

Das transições obtidas, as transições de transporte são separadas em um subgrupo:

$$T_\tau = \{T112, T125, T156, T245, T252, T223\}$$

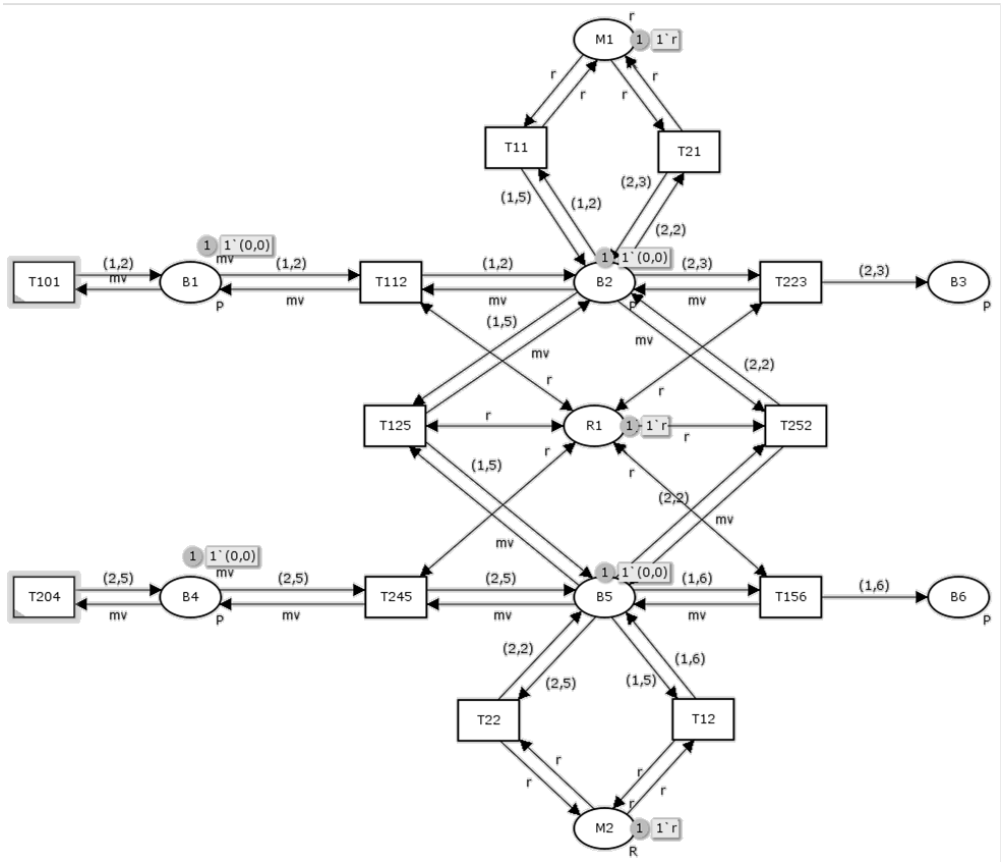


Figura 45: Modelo de CTPN resultante do cenário de exemplo

Para aplicar a técnica de resolução de *deadlock* é preciso adicionar novas informações ao conjunto de marcações, em outras palavras, é criado um novo conjunto de cor que compõe as marcações que representam os produtos.

$PRODUTO = \{p_i(t, e, rc), \ldots, p_n(t, e, rc)\}$ onde t representa o tipo do produto, e representa a etapa atual e rc representa a região crítica em que a marcação se encontra. Caso um produto não passe por qualquer região crítica, então, se um tipo de produto passe por duas ou mais regiões críticas, o valor de rc irá variar conforme a identificação da região que ele passe.

As transições que antecedem a região crítica só podem ser disparadas se as marcações da região crítica não pertencerem a essa região, ou seja, mesmo que uma marcação pertencente a região crítica entre na região o sistema não dará condições para a ocorrência de *deadlock*.

A evolução das marcações mostrada na Tabela 3, é o resultado do modelo gerado a partir da ordem de produção de um produto do tipo 1 e um produto do tipo 2. Tendo essas informações da ordem de produção e o modelo, é de conhecimento que a marcação final do modelo é apresentado na Tabela 2 (MF). Onde a coluna *Marcações* representa o conjunto de marcações correntes a partir de *M0*, a coluna *Disparo* representa qual transição foi

disparada para alcançar a marcação correspondente, as colunas *B1, B2, B3, B4, B5* e *B6* representam as marcações contidas nos *places B1, B2, B3, B4, B5* e *B6* respectivamente. Na Tabela 2 são exibidos somente os valores do tipo de produto nas marcações como base para as tabelas seguintes.

Tabela 2: Evolução das marcações de M0 à MF

Marcações	Disparo	B1	B2	B3	B4	B5	B6
M1	T101	(1)					
M2	T204	(1)			(2)		
...	...	...	...	...	...	...	...
MF	...			(2)			(1)

Observando a Tabela 3, a marcação final (*M6*) não corresponde à marcação final da Tabela 2 (MF). Na Figura 46 é apresentada a CPN na marcação *M6*, onde existe uma marcação (1,5) no *place B2* e uma marcação (2,2) no *place B5*. É observado na figura que as transições *T125* e *T252* estão desabilitadas, estas transições fazem o transporte entre os *places B2* e *B5* em sentidos opostos, em outras palavras, o conjunto de transições e *places* formam uma região crítica. Quando uma marcação chega a um dos *buffers* da região crítica (*B1* e *B2*), o sistema está em um estado que antecede uma condição de *deadlock* de espera circular.

Tabela 3: Evolução das marcações do modelo

Marcações	Disparo	B1	B2	B3	B4	B5	B6
M1	T101	(1,2)					
M2	T204	(1,2)			(2,5)		
M3	T112		(1,2)		(2,5)		
M4	T245		(1,2)			(2,5)	
M5	T11		(1,5)			(2,5)	
M6	T22		(1,5)			(2,2)	

Região Crítica: é o conjunto de *places* e transições tal que:

$RC_i = \{P_i, T_i\}$ onde $P_i = \{b_w, b_x, \dots, b_y, b_z\}$ e $T_i = \{t_{m,b_w,b_x}, \dots, t_{m,b_y,b_x}\}$ onde $m \in M$ e M é o conjunto de marcações da CPN. Em outras palavras, uma região crítica é um conjunto de transições e *places* que formam uma cadeia de requisições de forma cíclica. Uma região crítica deve conter um conjunto de dois ou mais *places* e um conjunto de duas ou mais transições.

O conjunto de transições que geram marcações a uma região crítica:

$T_{ARC_i} = \{t_{m,p,b_w}, t_{m,p,b_x}, \dots, t_{m,p,b_y}, t_{m,p,b_z}\}$ onde $p \in P$ e P é o conjunto de *places* da CPN.

No exemplo dado tem-se:

$RC_1 = \{P_1, T_1\}$ sendo que $P_1 = \{B1, B2\}$ e $T_1 = \{T125, T252\}$

e $T_{ARC_1} = \{T112, T245\}$.

Técnica de resolução de *deadlock:* Se uma marcação pertencente à uma região crítica estiver em um *place* dessa região, então nenhuma transição que gera marcações na região pode ser disparada até que a marcação saia da região crítica.

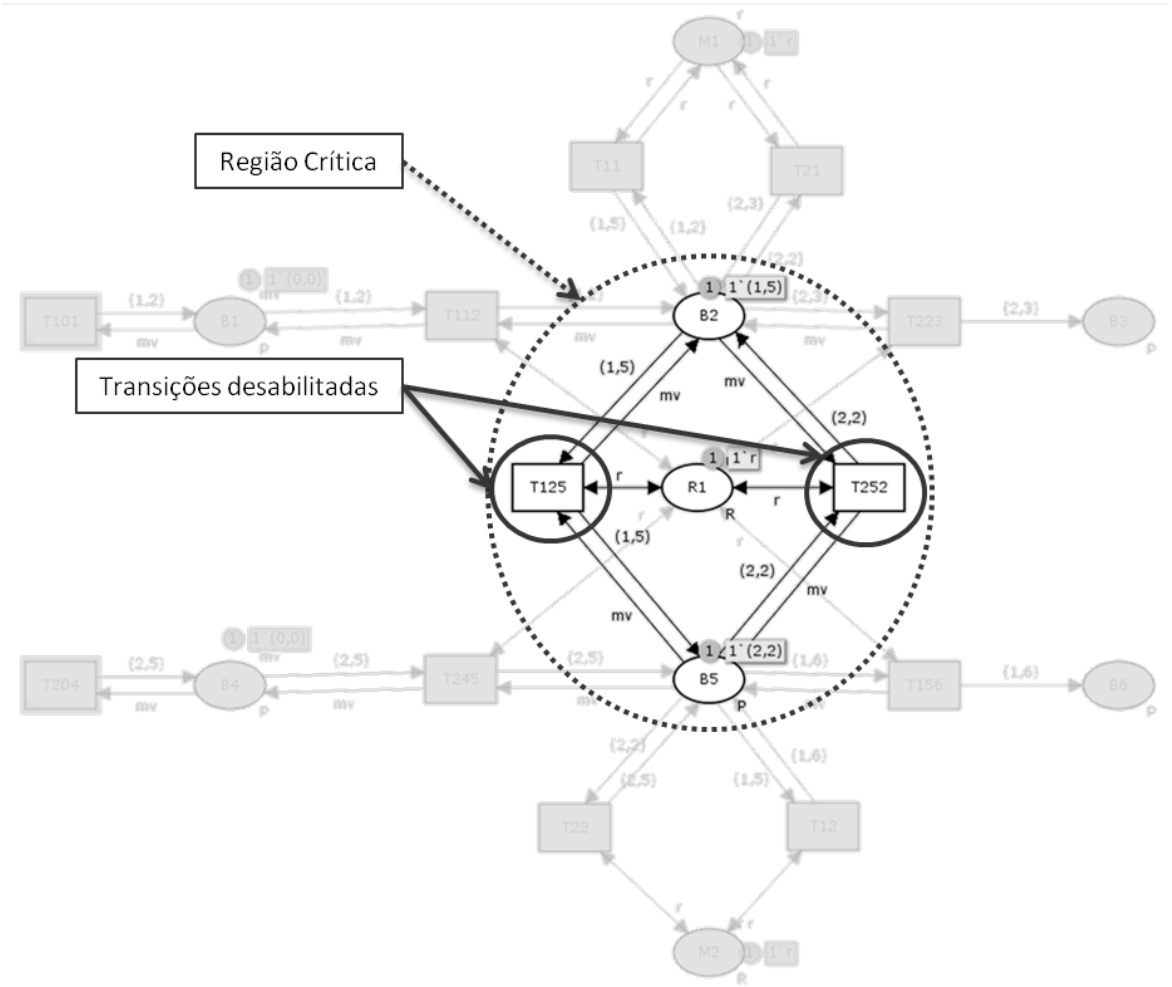


Figura 46: Estado de *deadlock* da CPN na marcação *M6*

Na Figura 47 é mostrado como a técnica é aplicada. Em destaque, a transição *T112* recebe um arco com a marcação correspondente às informações sobre o *buffer* que é representado pelo *place* $B5 \in P_1$, se a marcação não pertence à região crítica ($rc \neq 1$) a transição será habilitada, caso contrário esta será desabilitada até que a marcação saia da região crítica.

Seguindo o exemplo dado, a técnica é aplicada também na transição *T245*. Com isso as marcações e os arcos são alterados conforme o padrão proposto e o modelo de CPN resultante são apresentados na Figura 48.

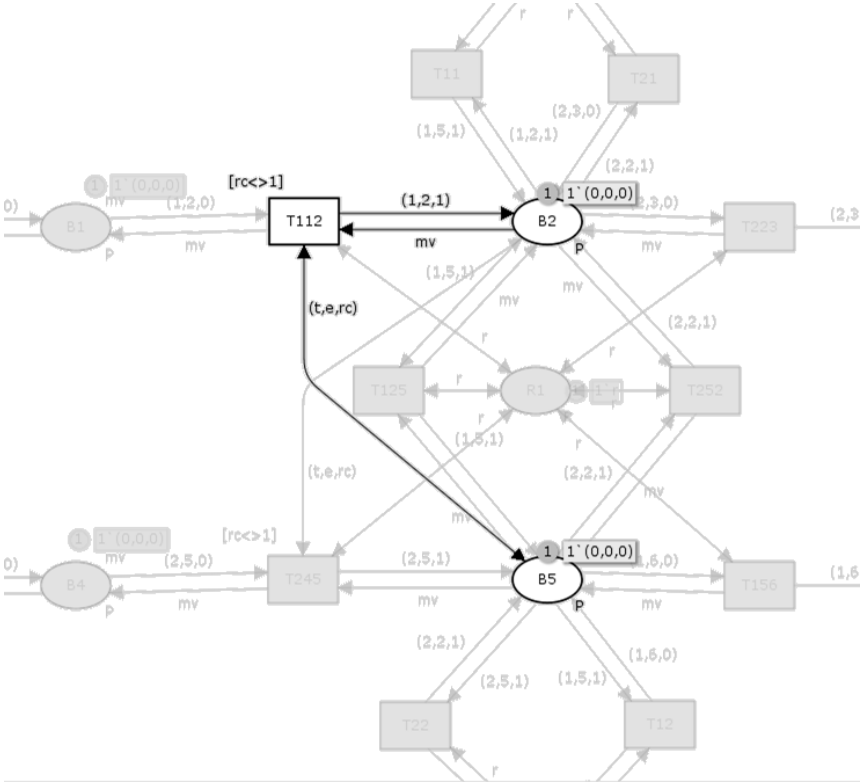


Figura 47: Transições que geram marcações na região crítica

Tabela 4: Evolução das marcações com a técnica aplicada

Marcações	Disparo	B1	B2	B3	B4	B5	B6
M1	T101	(1,2,0)					
M2	T201	(1,2,0)			(2,2,0)		
M3	T112		(1,2,1)		(2,2,0)		
M4	T11		(1,5,1)		(2,2,0)		
M5	T125				(2,2,0)	(1,5,1)	
M6	T12				(2,2,0)	(1,6,0)	
M7	T156				(2,2,0)		(1,6,0)
M8	T245					(2,5,1)	(1,6,0)
M9	T22					(2,2,1)	(1,6,0)
M10	T252		(2,2,1)				(1,6,0)
M11	T21		(2,3,0)				(1,6,0)
M12	T223			(2,3,0)			(1,6,0)

Observando a tabela de evolução das marcações com a técnica aplicada, nota-se que a marcação final (*M12*) corresponde à marcação final da Tabela 2 (*MF*). Comparando a

Tabela 2 com a Tabela 3 observa-se que depois que a transição *T112* é disparada no modelo com a técnica aplicada, a transição *T245* é desabilitada e só é habilitada depois do disparo da transição *T156*, diferente do que acontece com o modelo sem a técnica aplicada, que permite o disparo da transição *T245* e resultando em *deadlock* dois estados à frente.

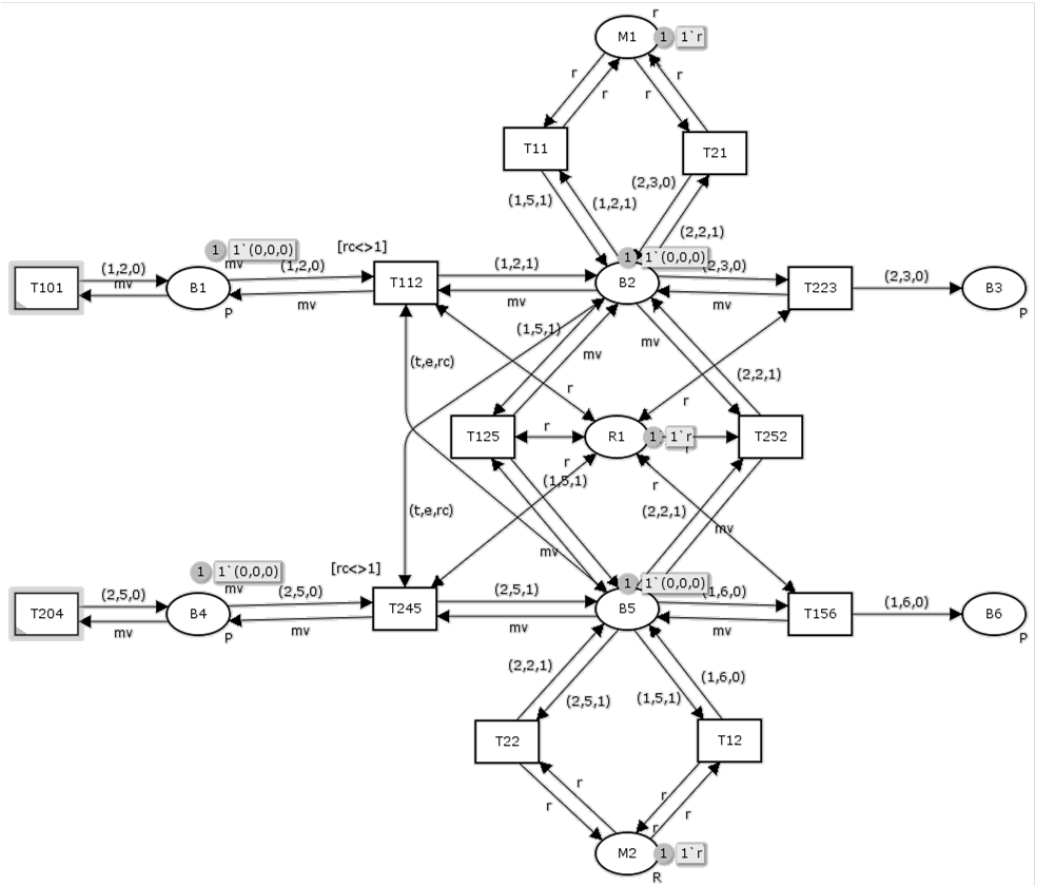


Figura 48: Modelo de CTPN do exemplo após a aplicação da técnica de DA proposta

4.6 Considerações finais

Neste capítulo foi descrito uma estratégia de resolução de *deadlock* para sistemas de controle de SMA baseada nas estratégias de *deadlock* estudadas e como é realizada a modelagem do SMA a qual é importante para que as condições de *deadlock* sejam definidas e a estratégia seja aplicada no modelo e no controle de um SMA

Em resumo o método consiste em:

1 – Estabelecer o modelo do SMA utilizando redes de Petri Coloridas Temporizadas de acordo com os modelos básicos citados orientados por recursos;

2 – Localizar as regiões críticas dentro do modelo estabelecido;

3 – Localizar as transições anteriores às transições do conjunto crítico;

4 – Inserir arcos orientados adicionais com informações atuais do chão de fábrica, como capacidade de *buffers*, prioridades de produtos.

Dessa forma esse método une características das estratégias de *deadlock prevention*, por permitir tratar uma modelagem já orientada à possível ocorrência de um deadlock e a sua solução, no entanto também trata de características da estratégia de *deadlock avoidance* devido a possibilidade do modelo ser utilizado para o controle do SMA e utilizar informações em tempo real do chão de fábrica para a resolução de um provável *deadlock*.

As definições descritas neste capítulo serão utilizadas no Capítulo 5 onde a técnica será implementada e validada.

Capítulo 5

IMPLEMENTAÇÃO

5.1 Considerações Iniciais

O capítulo anterior apresentou a proposta deste trabalho. A partir das regras de modelagem e da técnica de resolução de *deadlock* definidas, pode-se gerar modelos de CPN determinísticos para o controle da produção. Neste capítulo será apresentada a fase de implementação e validação da proposta, utilizando dois exemplos de cenários apresentados anteriormente. No cenário proposto por Piroddi, Cordone e Fumagalli (2008), será apresentada uma análise comportamental do modelo gerado. No cenário proposto por Roszkowska (2004) e adaptado por Wu, Zhou e Li (2008), será apresentada uma análise mais aprofundada, utilizando análise de relação transição/evento. É Importante observar que os cenários possuem nomenclaturas distintas para alguns tipos de recursos ou elementos do cenário. Esta divergência de padrão de nomes se dá devido aos diferentes arranjos e seguimentos para os quais cada cenário é proposto. Sendo assim é importante notar previamente a nomenclatura de cada cenário para melhor compreensão das figuras apresentadas.

5.1.1 Aplicação e análise comportamental dos resultados

O primeiro sistema utilizado como exemplo para a implementação e validação deste trabalho foi proposto por Piroddi, Cordone e Fumagalli (2008).

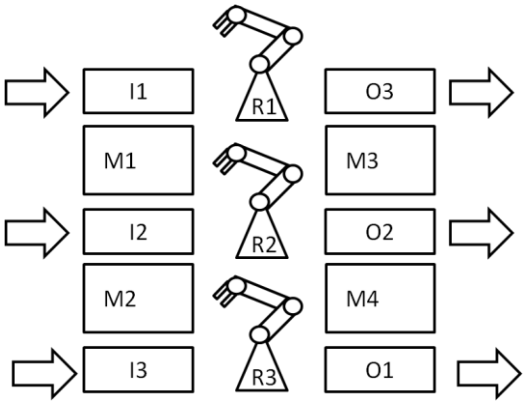


Figura 49: Arranjo do cenário proposto por Piroddi, Cordone e Fumagalli (2008).

O cenário consiste em quatro estações de trabalho (M1, M2, M3 e M4), dez *buffers,* sendo quatro das estações de trabalho (B1, B2, B3 e B4), três *buffers* de entrada (I1, I2 e I3) e três *buffers* de saída (O1, O2 e O3), três robôs de movimentação (R1, R2 e R3) e três tipos de produtos (P1, P2 e P3). Na Figura 49 são apresentados os três roteiros de produção.

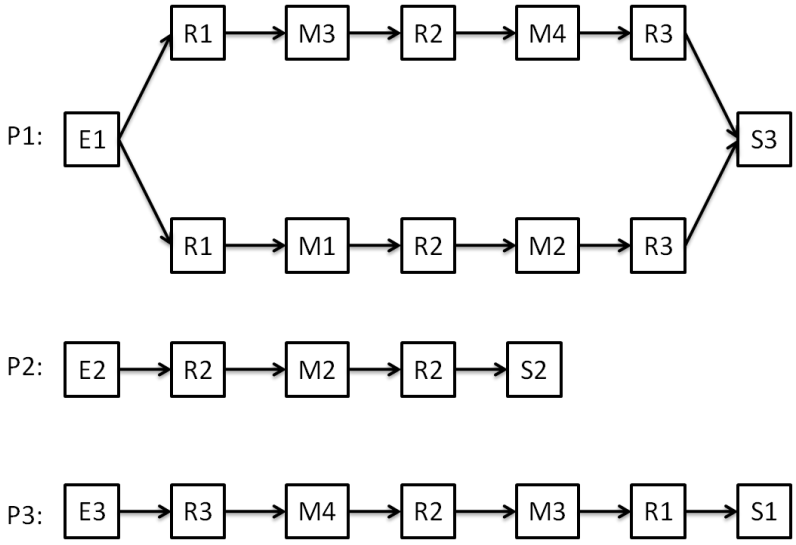


Figura 50: Roteiros de produção dos produtos P1, P2 e P3.

É possível notar na Figura 50 que somente os produtos do tipo P1 possuem rotas alternativas de produção, neste caso, se em algum momento da produção o recurso de uma das rotas estiver indisponível o controlador da produção pode trocar a rota do produto em tempo de operação.

A partir do arranjo e rota de produção dados, é possível, a partir do método proposto no Capítulo 4, modelar uma rede de Petri colorida para o controle da produção do cenário. Para que a análise do comportamento seja feita, foram atribuídas ordens de produção. Uma

dessas ordens de produção será considerada para essa análise. Na ordem de produção apresentada na Tabela 5, um produto do tipo *P1* entrará no cenário no tempo zero, em seguida, no tempo 1, um produto do tipo *P1* e um do tipo *P3* entrará no cenário. No tempo 2 entrará um produto do tipo *P2* e nos tempos 300 e 605 entrará um produto do tipo *P2* e um do tipo *P3* respectivamente.

Tabela 5: Ordem de produção

Tempo	Tipo de Produto
0	P1
1	P1, P3
2	P2
300	P2
605	P3

Na Figura 51 é apresentado o resultado da modelagem para o cenário proposto considerando a ordem de produção. Comparando a Figura 51 com a Figura 49, é possível notar algumas semelhanças, como a disposição dos elementos do cenário. Os únicos elementos que não aparecem na Figura 49 são os buffers de entrada e saída das máquinas (*B1*, *B2*, *B3* e *B4*).

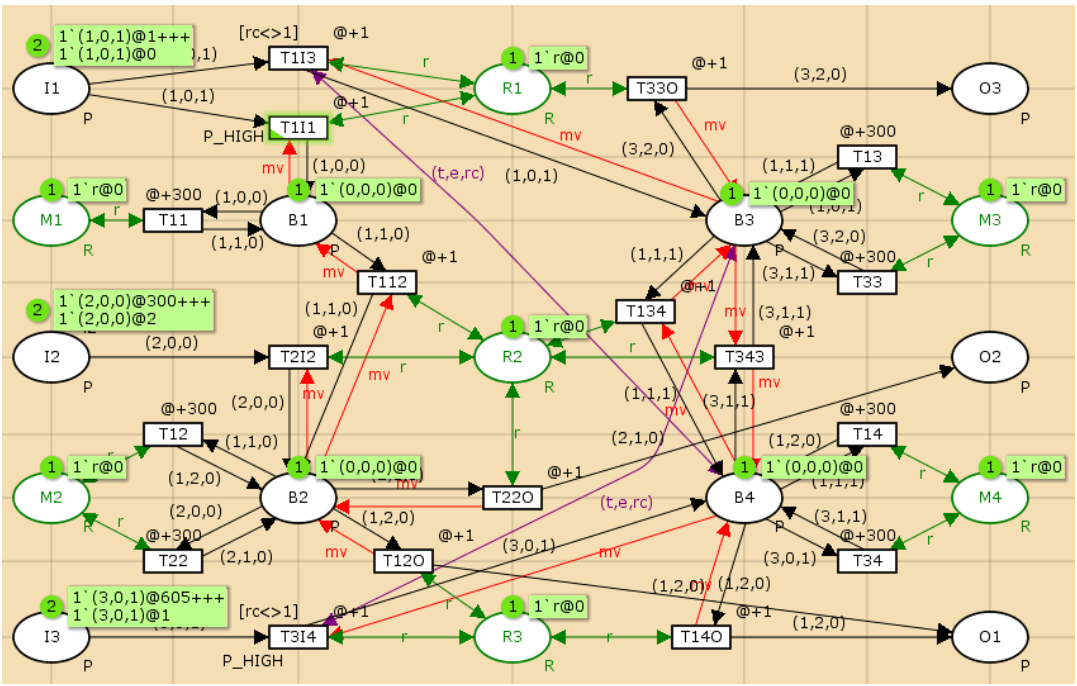
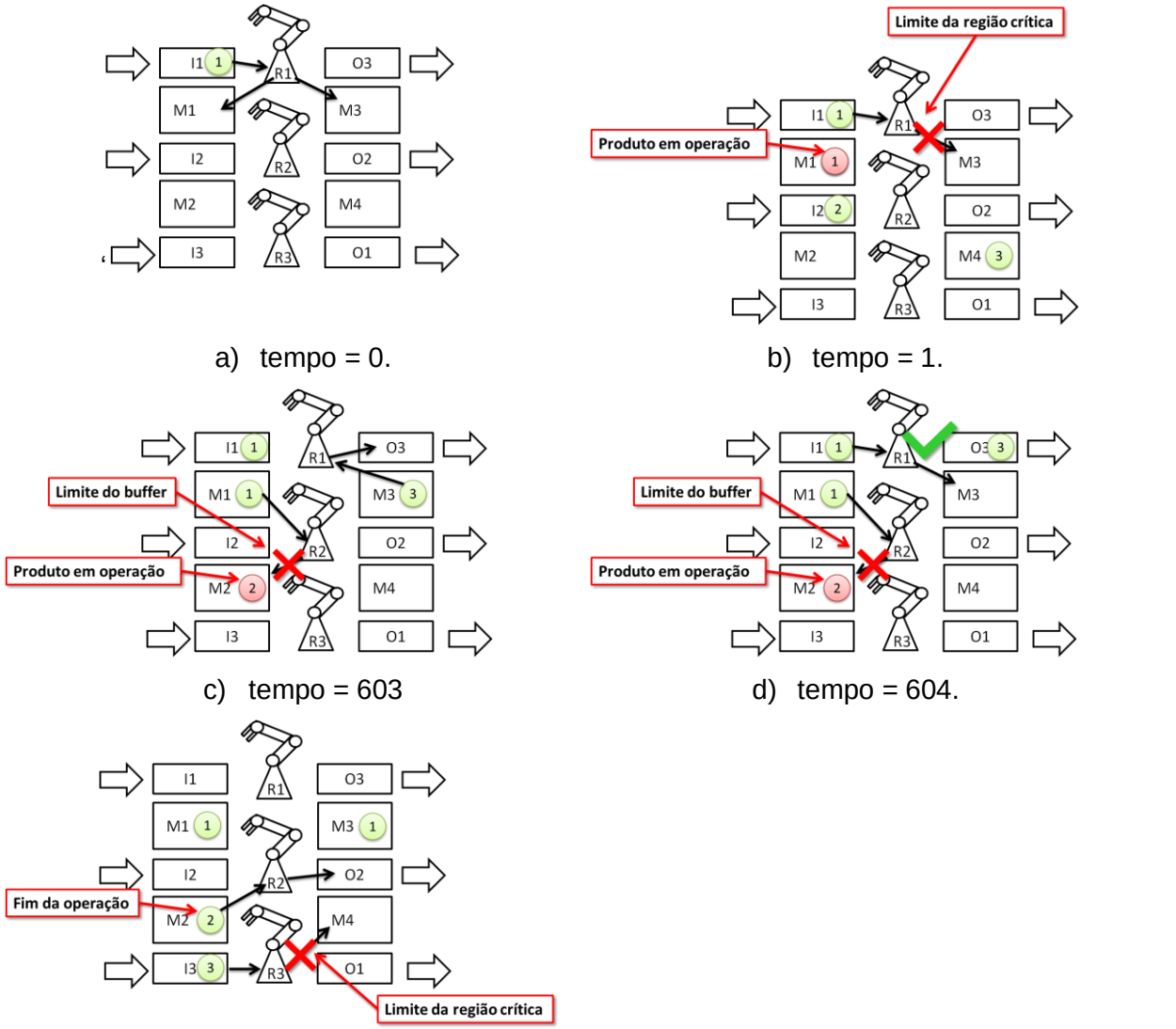


Figura 51 : Modelo de rede de Petri Colorida gerado.

O comportamento do sistema após a aplicação do modelo gerado foi simulado e ilustrado na Figura 52.



a) tempo = 0.

b) tempo = 1.

c) tempo = 603

d) tempo = 604.

e) tempo = 605

Figura 52: Ilustração do comportamento do FMS dada a ordem de produção da Tabela 5.

No tempo zero (ver Figura 52 a) um produto do tipo 1 entra no sistema para ser processado através da entrada “I1”, neste momento as transições “T1I3” e “T1I1” são habilitadas para serem disparadas. Como qualquer uma das duas transições não resultarão em *deadlocks*, qualquer uma das transições pode ser disparada, neste caso, o CPN Tools escolheu arbitrariamente disparar a transição “T1I1” que enviará o comando para o robô “R1” transportar o produto para a máquina “M1”.

Na Figura 52 b, é representada a situação do FMS no tempo 1, onde o produto que entrou anteriormente está em produção e dois novos produtos do tipo 1 e tipo 3 entram para

serem produzidos. Nesta situação, o produto do tipo 1 não pode ser processado na máquina “M1”, restando a segunda rota, ao mesmo tempo, o produto do tipo 3 está pronto para ser produzido na máquina “M4”. Neste caso, tanto o produto do tipo “P1” quanto o produto do tipo “P3” estão prestes a entrar em uma região crítica, pois segundo o roteiro de produção de “P1”, serão utilizadas as máquinas “M3” e “M4” sequencialmente, enquanto no roteiro do produto “P3” serão utilizadas as máquinas “M4” e “M3” sequencialmente. Para evitar uma situação de *deadlock* o controlador da produção que está seguindo o modelo proposto só irá autorizar uma das duas transições disponíveis (ou “T1I3” ou “T3I4”), assim que uma das transições for disparada a outra será desabilitada. Na simulação, a transição “T3I4” foi disparada e a transição “T1I3” foi desabilitada até que seja liberada a máquina “M1” ou a região crítica.

No tempo 603 (Figura 52c) um produto do tipo “P2” está em operação na máquina “M2” enquanto isso, a máquina “M1” finalizou a operação do produto “P1”, mas o mesmo não pode prosseguir enquanto o produto do tipo “P2” não liberar “M2”. No mesmo tempo a máquina “M3” finalizou a produção do produto do tipo “P3” e será enviado um comando para o robô “R1” fazer a movimentação de “P3” da “M3” para “O3”.

No tempo 604 (Figura 52d) o robô “R1” concluiu a operação de movimentação do produto do tipo “P3” para a saída “O3”, com o produto do tipo “P1” que estava em “I1” foi liberado para ser transportado para “M3”.

No tempo 605 (Figura 52e) a produção do produto do tipo “P2” finalizou, portanto, assim que a estação “M2” for liberada, o produto do tipo “P1” que está em “M1” será liberado para ser transportado até “M2”. Ao mesmo tempo, um produto do tipo “P3” entrou no sistema pela entrada “I3”, porém sua movimentação para “M4” não será autorizada até que o produto do tipo “P1” libere a região crítica.

Como resultado, não tiveram situações de *deadlock* no sistema durante a simulação, na rede de Petri gerada todas as transações foram disparadas até que a marcação final fosse alcançada, e o controlador permitiu o máximo de recursos possíveis em operação simultaneamente.

5.2 Aplicação e análise formal dos resultados

O segundo sistema utilizado como exemplo para implementação e validação deste trabalho foram proposto por Roszkowska (2004) e utilizado e adaptado por Wu, Zhou e Li

(2008). O sistema é um FAS composto por dois robôs de manuseio r1 e r2, e três estações de trabalho (w1, w2 e w3). Tal sistema já foi descrito na no capítulo 3, seção 3.4.2.

As Figuras 13 e 14 ilustram o problema, além das características descritas, foi adicionada mais uma, para que a simulação seja feita em um modelo de CPN, que representa o fator tempo. Nota-se que o cenário descrito não possui informações sobre os tempos de operação e transporte das peças. Portanto, foram definidos os seguintes valores: todos os eventos de transportes demoram uma unidade de tempo (u.t.) para serem concluídos. As operações de produção da estação w1 demoram quatrocentas unidades de tempo (400 u.t.) e as estações de produção w2 e w3 levam trezentas unidades de tempo para efetuarem as operações de produção tanto dos produtos tipo-A quanto tipo-B.Com base nas informações obtidas do cenário e nas regras de modelagem desta proposta, tem-se o modelo de CTPN resultante como é mostrado na Figura 61. Os eventos de operações de produção da estação w_1 para os produtos de tipo-A e B são representados pelas transições *T11* e *T21* respectivamente, como é mostrado na Figura 53. Os eventos que representam a operação de produção dos produtos de tipo-A e B para a estação de produção w_2 são representados pelas transições *T12* e *T22* como é mostrado na Figura 54. E as transições *T13* e *T23* representam as operações de produção dos produtos de tipo-A e B pela estação de produção w_3 respectivamente como é apresentado na Figura 55.

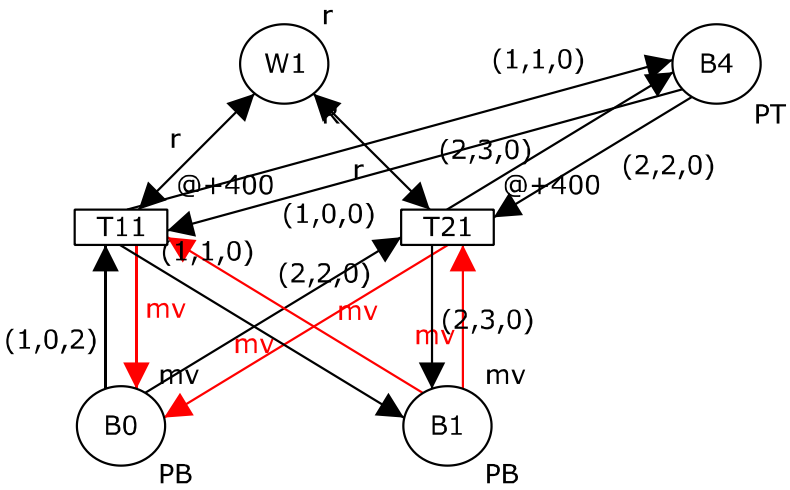


Figura 53: Elementos relacionados à estação de trabalho w_1

Os *places* W1-3 simulam a disponibilidade das estações de produção w_1, w_2 e w_3 respectivamente, no SMA. Os *places B0-5* representam os *buffers* do sistema, onde cada um tem sua limitação conforme as características descritas anteriormente.

Na Figura 56 são mostrados os elementos que representam os eventos disponíveis para o robô de movimentação r_1 e na Figura 57 para o robô r_2.

As marcações de tipo 1 e 2 representam os produtos de tipo-A e B que podem ser produzidos no FMS em questão.

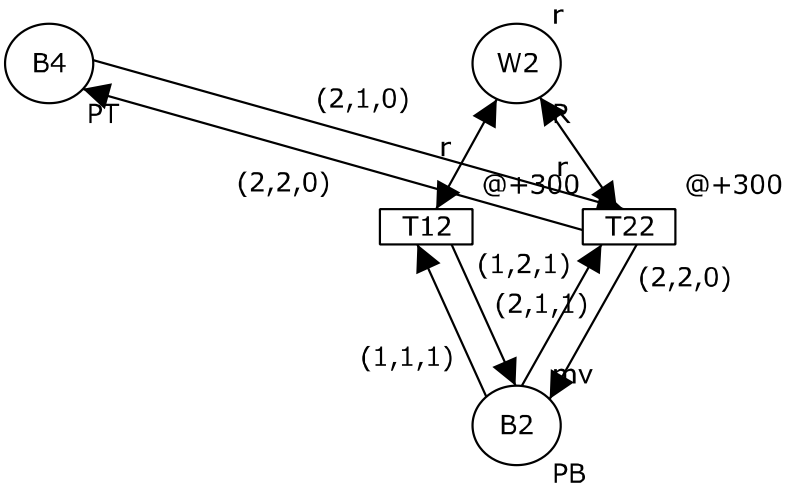


Figura 54: Elementos relacionados à estação de trabalho w_2

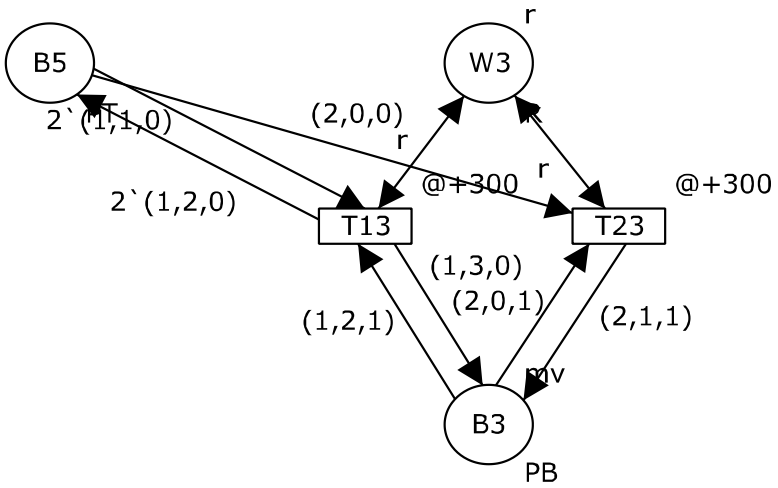


Figura 55: Elementos relacionados à estação de trabalho w_3

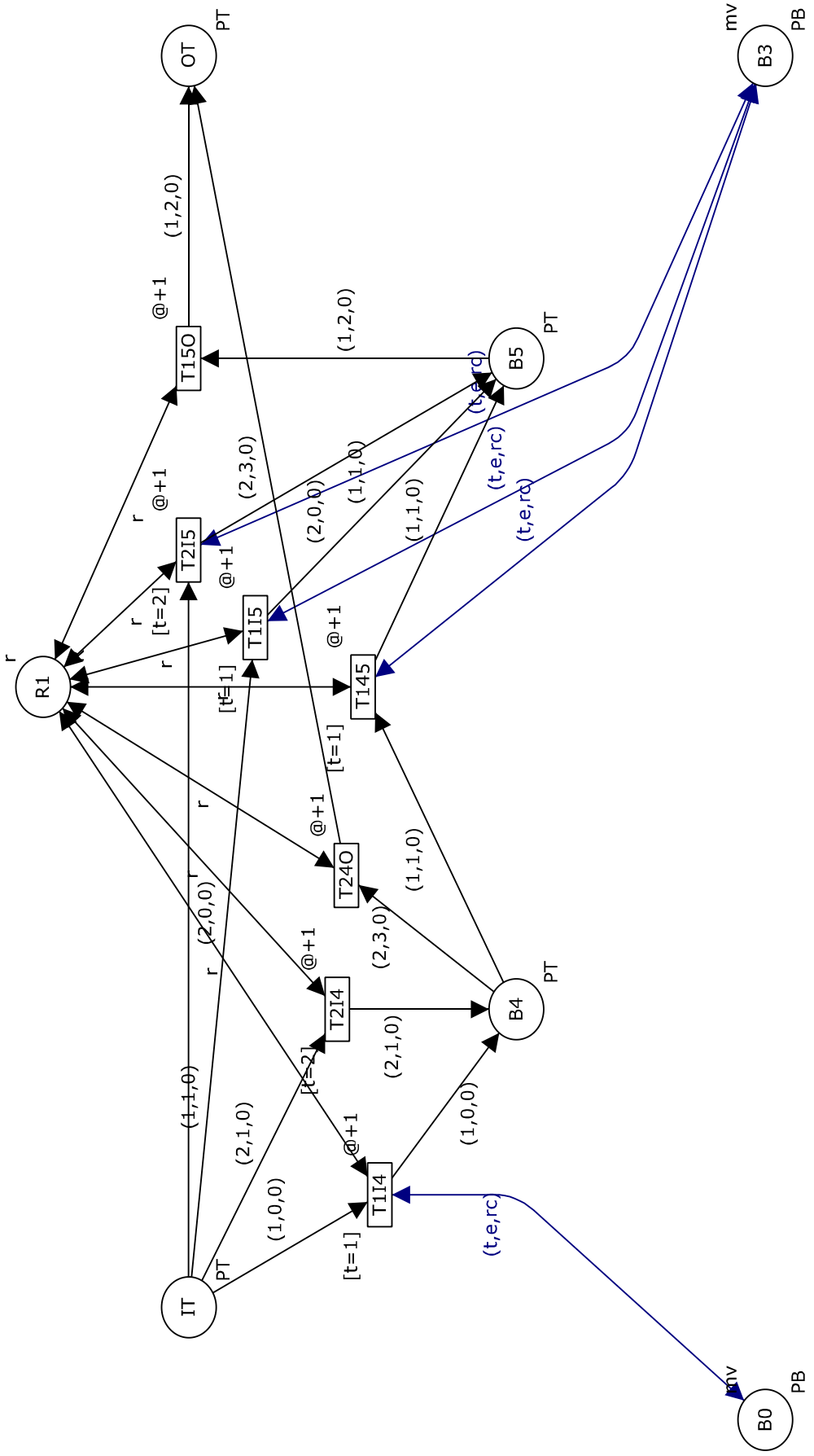


Figura 56: Elementos relacionados ao robô r_1

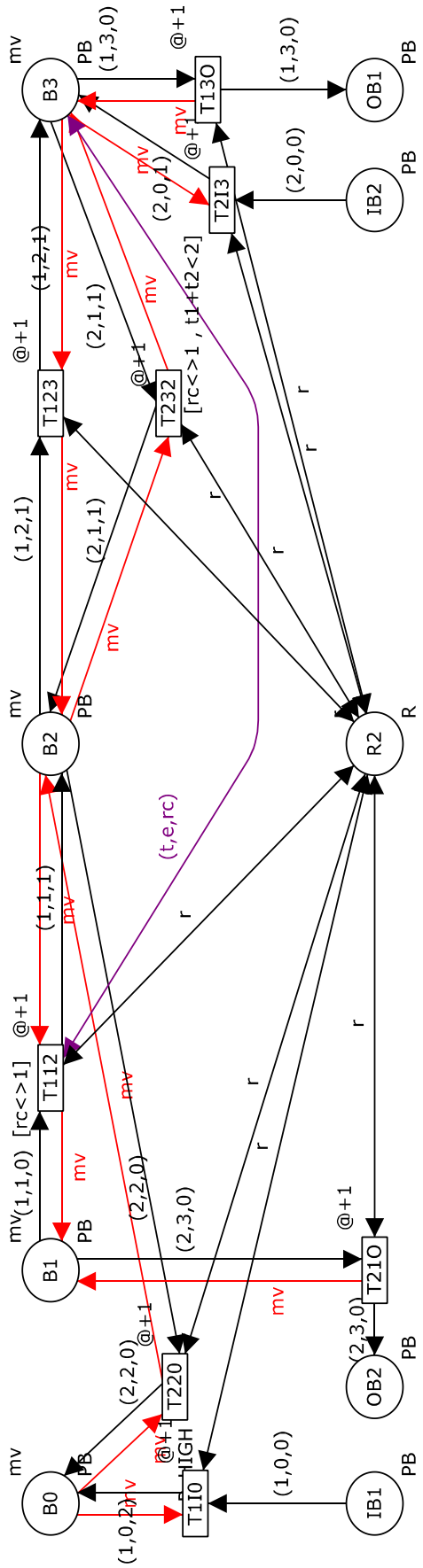


Figura 57: Elementos relacionados ao robô r_2

A partir de uma busca de estados de *deadlock* no sistema, foi possível identificar, inicialmente uma região crítica onde pode existir um impasse caso uma marcação do tipo 1 se encontre no *place B2* e uma marcação do tipo 2 esteja no *place B3*, como é ilustrado na Figura 58.

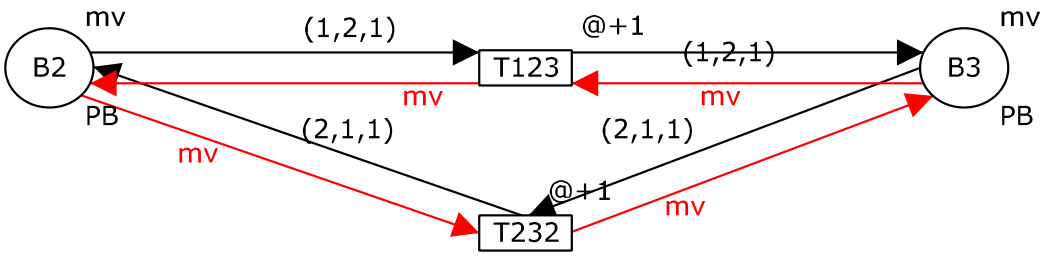


Figura 58: Região Crítica 1

Além disso, observou-se também que no conjunto dos *places* $b_{0\text{-}3}$ não pode haver duas marcações do tipo 1 e uma do tipo 2 ao mesmo tempo e também não pode haver duas marcações do tipo 2 e uma do tipo 1 ao mesmo tempo, em qualquer ordem. Isto torna toda a região uma segunda região crítica, como é mostrada na Figura 59.

No entanto, são inseridos arcos às regiões críticas conforme a técnica de resolução de *deadlock* desta proposta. Na Figura 59 é apresentada a disposição dos arcos de controle na CTPN gerada.

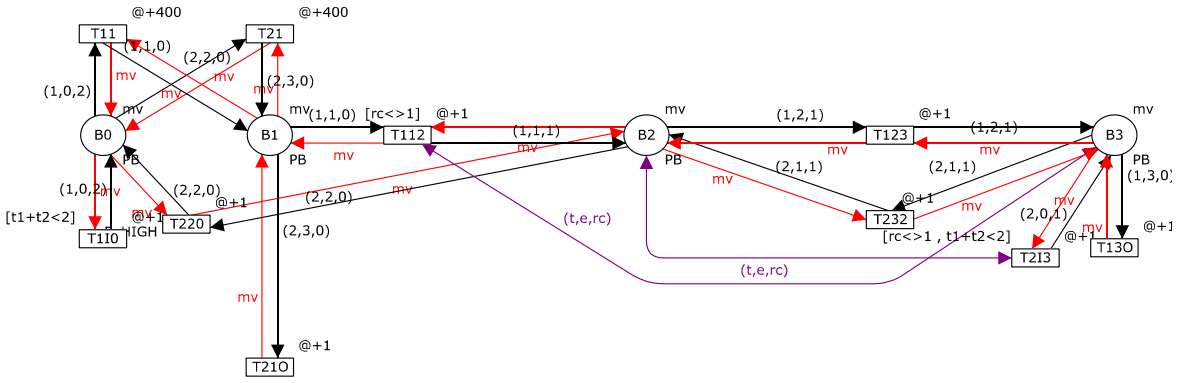


Figura 59: Região Crítica 2

```
Conjunto de cores
      colset INT = int with 0..10;
      colset R = int with r timed;
      colset T = int with 0..2;
      colset E = int with 0..4;
      colset RC = int with 0..2;
      colset TxT = product INT*INT;
      colset PT = product T*E*RC timed;
      colset PB = product T*E*RC timed;
Variáveis
      var t: T;
      var t1,t2: INT;
      var e: E;
      var RC: rc;
Constantes
      val mv = (0,0,0);
Funções
      fun I(t,t1,t2) = if t=1 then (t1+1,t2) else (t1,t2+1);
      fun D(t,t1,t2) = if t=1 then (t1-1,t2) else (t1,t2-1);
```

Figura 60: Conjuntos de cores, variáveis, constantes e funções

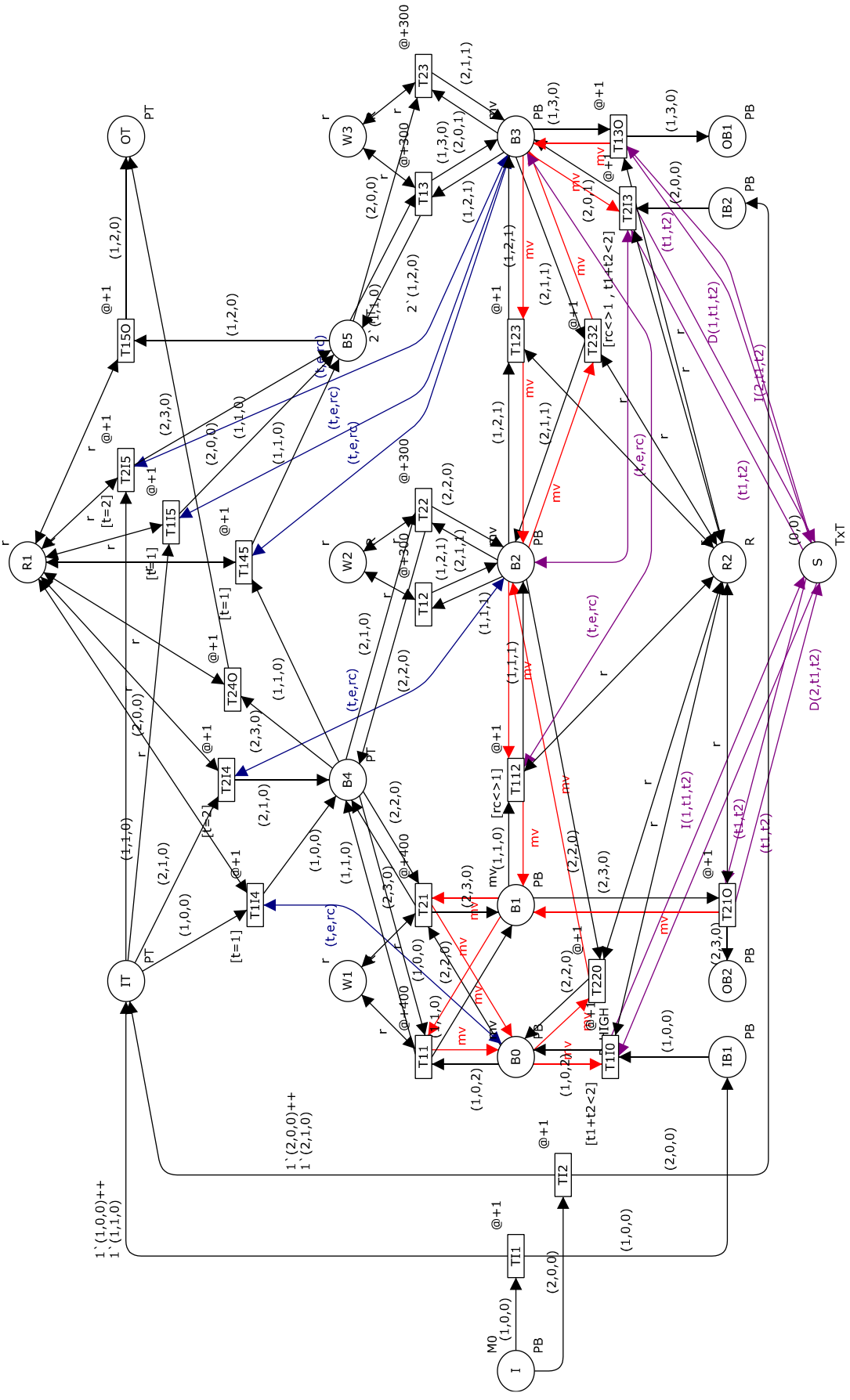


Figura 61: Modelo de CTPN resultante

Os conjuntos de cores são representados na ferramenta de modelagem como conjuntos numéricos, assim como na Figura 60. Onde cada "*colset"* é um conjunto de cores.

5.3 Teste de validação

Depois que foi gerado o modelo de CTPN, o processo de validação foi iniciado, o processo consiste em verificar a integridade da técnica aplicada à modelagem.

Inicialmente, foram definidos todos os possíveis estados de *deadlock* do sistema. A partir desses estados, foram geradas marcações iniciais de forma que se não houvesse uma estratégia de resolução de *deadlock* aplicada, certamente o sistema iria entrar em estado de *deadlock*. Se a simulação do sistema não entrar em estado de *deadlock* significa que a rede é livre de *deadlock*.

Os possíveis estados de *deadlock* do sistema são os seguintes:

Estado de *deadlock* 1*:* Se existe uma marcação do tipo 1 no *place B2* e no mesmo momento existe uma marcação do tipo 2 no *place B3* então o sistema está em estado de *deadlock, pois* a única transição que consome a marcação do tipo 1 do *place B2* requer o *place B3* vazio, da mesma forma, a transição que consome a marcação do tipo 2 do *place B3*, requer o *B2* vazio.

Estado de *deadlock* 2: Se existe uma marcação do tipo 1 em *B0*, uma marcação do tipo 1 em *B1* e uma marcação do tipo 2 em *B2*, então o sistema está em *deadlock*. Pois existe uma condição de espera circular entre os três *places*.

Estado de *deadlock* 3: Se existe uma marcação do tipo 2 em *B0*, uma marcação do tipo 1 em *B1* e uma marcação do tipo 2 em *B2*, então o sistema está em estado de *deadlock* pela condição de espera circular.

Além desses estados de *deadlock,* existem outros estados que não estão em *deadlock*, mas são antecessores aos estados de *deadlock* e que a partir desses, não há outros estados sucessores que não são os de *deadlock*, logo, para que o sistema seja livre de *deadlock* esses estados também devem ser evitados.

Uma rede de Petri é dita livre de *deadlock* se esta parte de um estado inicial M_0 e alcança um dos estados finais, sendo que qualquer estado que a rede possa alcançar não ocorra *deadlock*.

Observando os resultados das simulações realizadas mostrados nas Tabelas 5-11, pode-se verificar que em nenhum momento o sistema entrou em estado de *deadlock*. Ou

seja, todas as peças foram montadas e as transições que levam as marcações para os *places* finais foram disparadas.

Também foi possível observar que o modelo do controlador permitiu um número máximo de peças em operação simultaneamente no sistema de forma que não ocorressem condições para a ocorrência de *deadlock*.

As tabelas a seguir são resultados das saídas das simulações feitas no modelo de CTPN gerado pela estratégia de resolução da proposta. Cada simulação foi feita com diferentes entradas, ou seja, diferentes ordens de produção durante a simulação.

Na tabela a seguir, é mostrado o que representa cada transição, em outras palavras, quais eventos do sistema de manufatura cada uma representa.

Tabela 5: Relação Transição/Evento do SMA

Transição	Tipo	Peça	Recurso	Origem	Destino	Duração do evento
T1I0	Movimentação	Base-A	Robô 2	Entrada	buffer 0	1
T11	Operação	Base-A	Estação 1	buffer 0	buffer 1	400
T112	Movimentação	Base-A	Robô 2	buffer 1	buffer 2	1
T12	Operação	Base-A	Estação 2	buffer 2	buffer 2	1
T123	Movimentação	Base-A	Robô 2	buffer 2	buffer 3	1
T13	Operação	Base-A	Estação 3	buffer 3	buffer 3	300
T13O	Movimentação	Base-A	Robô 2	buffer 3	Saída	1
T1I4	Movimentação	Tray-A	Robô 1	Entrada	buffer 4	1
T145	Movimentação	Tray-A	Robô 1	buffer 4	buffer 5	1
T1I5	Movimentação	Tray-A	Robô 1	Entrada	buffer 5	1
T15O	Movimentação	Tray-A	Robô 1	buffer 5	Saída	1
T2I3	Movimentação	Base-B	Robô 2	Entrada	buffer 3	1
T23	Operação	Base-B	Estação 3	buffer 3	buffer 3	300
T232	Movimentação	Base-B	Robô 2	buffer 3	buffer 2	1
T22	Operação	Base-B	Estação 2	buffer 2	buffer 2	300
T220	Movimentação	Base-B	Robô 2	buffer 2	buffer 0	1
T21	Operação	Base-B	Estação 1	buffer 0	buffer 1	400
T21O	Movimentação	Base-B	Robô 2	buffer 1	Saída	1
T2I4	Movimentação	Tray-B	Robô 1	Entrada	buffer 4	1
T2I5	Movimentação	Tray-B	Robô 1	Entrada	buffer 5	1
T24O	Movimentação	Tray-B	Robô 1	buffer 4	Saída	1

Simulação 1

Entrada: um produto tipo-A

Tabela 6: Saída da simulação 1

Step	u.t	Tansição Disparada	Recurso	Evento	Peça	Destino
1	1	T1I0	Robô 2	Movimentação	Base-A	buffer 0
2	2	T1I4	Robô 1	Movimentação	Tray-A	buffer 4
3	3	T11	Estação 1	Operação	Base-A	buffer 1
4	403	T112	Robô 2	Movimentação	Base-A	buffer 2
5	404	T12	Estação 2	Operação	Base-A	buffer 2
6	704	T123	Robô 2	Movimentação	Base-A	buffer 3
7	705	T1I5	Robô 1	Movimentação	Tray-A	buffer 5
8	706	T145	Robô 1	Movimentação	Tray-A	buffer 5
9	707	T13	Estação 3	Operação	Base-A	buffer 3
10	1007	T15O	Robô 1	Movimentação	Tray-A	Saída
11	1007	T13O	Robô 2	Movimentação	Base-A	Saída
12	1008	T15O	Robô 1	Movimentação	Tray-A	Saída

Simulação 2

Entrada: um produto tipo-B

Tabela 7: Saída da simulação 2

Step	u.t	Tansição Disparada	Recurso	Evento	Peça	Destino
1	1	T2I3	Robô 2	Movimentação	Base-B	buffer 3
2	2	T2I5	Robô 1	Movimentação	Tray-B	buffer 5
3	3	T23	Estação 3	Operação	Base-B	buffer 3
4	303	T232	Robô 2	Movimentação	Base-B	buffer 2
5	304	T2I4	Robô 1	Movimentação	Tray-B	buffer 4
6	305	T22	Estação 2	Operação	Base-B	buffer 2
7	605	T220	Robô 2	Movimentação	Base-B	buffer 0
8	606	T21	Estação 1	Operação	Base-B	buffer 1
9	1006	T24O	Robô 1	Movimentação	Tray-B	Saída
10	1006	T21O	Robô 2	Movimentação	Base-B	Saída

Simulação 3

Entrada: um produto tipo-A e um produto tipo-B

Tabela 8: : Saída da simulação 3

Step	u.t	Tansição Disparada	Recurso	Evento	Peça	Destino
1	1	T2I3	Robô 2	Movimentação	Base-B	buffer 3
2	2	T1I0	Robô 2	Movimentação	Base-A	buffer 0
3	2	T2I5	Robô 1	Movimentação	Tray-B	buffer 5
4	3	T23	Estação 3	Operação	Base-B	buffer 3
5	3	T1I4	Robô 1	Movimentação	Tray-A	buffer 4
6	4	T11	Estação 1	Operação	Base-A	buffer 1
7	303	T232	Robô 2	Movimentação	Base-B	buffer 2
8	304	T2I4	Robô 1	Movimentação	Tray-B	buffer 4
9	305	T22	Estação 2	Operação	Base-B	buffer 2
10	605	T220	Robô 2	Movimentação	Base-B	buffer 0
11	606	T112	Robô 2	Movimentação	Base-A	buffer 2
12	607	T21	Estação 1	Operação	Base-B	buffer 1
13	607	T12	Estação 2	Operação	Base-A	buffer 2
14	907	T123	Robô 2	Movimentação	Base-A	buffer 3
15	908	T1I5	Robô 1	Movimentação	Tray-A	buffer 5
16	909	T145	Robô 1	Movimentação	Tray-A	buffer 5
17	910	T13	Estação 3	Operação	Base-A	buffer 3
18	1007	T21O	Robô 2	Movimentação	Base-B	Saída
19	1007	T24O	Robô 1	Movimentação	Tray-B	Saída
20	1210	T13	Estação 3	Operação	Base-A	buffer 3
21	1210	T15O	Robô 1	Movimentação	Tray-A	Saída
22	1211	T15O	Robô 1	Movimentação	Tray-A	Saída

Simulação 4

Entrada: um produto tipo-A e dois produtos tipo-B

Tabela 9: Saída da simulação 6

Step	u.t	Tansição Disparada	Recurso	Evento	Peça	Destino
1	1	T1I0	Robô 2	Movimentação	Base-A	buffer 0
2	2	T1I4	Robô 1	Movimentação	Tray-A	buffer 4
3	2	T2I3	Robô 2	Movimentação	Base-B	buffer 3
4	3	T11	Estação 1	Operação	Base-A	buffer 1
5	3	T2I5	Robô 1	Movimentação	Tray-B	buffer 5
6	4	T23	Estação 3	Operação	Base-B	buffer 3
7	304	T232	Robô 2	Movimentação	Base-B	buffer 2
8	305	T2I4	Robô 1	Movimentação	Tray-B	buffer 4
9	306	T22	Estação 2	Operação	Base-B	buffer 2
10	606	T220	Robô 2	Movimentação	Base-B	buffer 0
11	607	T112	Robô 2	Movimentação	Base-A	buffer 2
12	608	T21	Estação 1	Operação	Base-B	buffer 1
13	608	T12	Estação 2	Operação	Base-A	buffer 2
14	908	T123	Robô 2	Movimentação	Base-A	buffer 3
15	909	T1I5	Robô 1	Movimentação	Tray-A	buffer 5
16	910	T145	Robô 1	Movimentação	Tray-A	buffer 5
17	911	T13	Estação 3	Operação	Base-A	buffer 3
18	1008	T24O	Robô 1	Movimentação	Tray-B	Saída
19	1008	T21O	Robô 2	Movimentação	Base-B	Saída
20	1211	T13O	Robô 2	Movimentação	Base-A	Saída
21	1211	T15O	Robô 1	Movimentação	Tray-A	Saída
22	1212	T15O	Robô 1	Movimentação	Tray-A	Saída
23	1212	T2I3	Robô 2	Movimentação	Base-B	buffer 3
24	1213	T2I5	Robô 1	Movimentação	Tray-B	buffer 5
25	1214	T23	Estação 3	Operação	Base-B	buffer 3
26	1514	T232	Robô 2	Movimentação	Base-B	buffer 2
27	1515	T2I4	Robô 1	Movimentação	Tray-B	buffer 4
28	1516	T22	Estação 2	Operação	Base-B	buffer 2
29	1816	T220	Robô 2	Movimentação	Base-B	buffer 0
30	1817	T21	Estação 1	Operação	Base-B	buffer 1
31	2217	T21O	Robô 2	Movimentação	Base-B	Saída
32	2217	T24O	Robô 1	Movimentação	Tray-B	Saída

Simulação 5

Entrada: dois produtos tipo-A e um produto tipo-B

Tabela 10: Saída da simulação 5

Step	u.t	Tansição Disparada	Recurso	Evento	Peça	Destino
1	1	T1I0	Robô 2	Movimentação	Base-A	buffer 0
2	2	T2I3	Robô 2	Movimentação	Base-B	buffer 3
3	2	T1I4	Robô 1	Movimentação	Tray-A	buffer 4
4	3	T11	Estação 1	Operação	Base-A	buffer 1
5	3	T2I5	Robô 1	Movimentação	Tray-B	buffer 5
6	4	T23	Estação 3	Operação	Base-B	buffer 3
7	304	T232	Robô 2	Movimentação	Base-B	buffer 2
8	305	T2I4	Robô 1	Movimentação	Tray-B	buffer 4
9	306	T22	Estação 2	Operação	Base-B	buffer 2
10	606	T220	Robô 2	Movimentação	Base-B	buffer 0
11	607	T112	Robô 2	Movimentação	Base-A	buffer 2
12	608	T21	Estação 1	Operação	Base-B	buffer 1
13	608	T12	Estação 2	Operação	Base-A	buffer 2
14	908	T123	Robô 2	Movimentação	Base-A	buffer 3
15	909	T145	Robô 1	Movimentação	Tray-A	buffer 5
16	910	T1I5	Robô 1	Movimentação	Tray-A	buffer 5
17	911	T13	Estação 3	Operação	Base-A	buffer 3
18	1008	T21O	Robô 2	Movimentação	Base-B	Saída
19	1008	T24O	Robô 1	Movimentação	Tray-B	Saída
20	1009	T1I0	Robô 2	Movimentação	Base-A	buffer 0
21	1010	T1I4	Robô 1	Movimentação	Tray-A	buffer 4
22	1011	T11	Estação 1	Operação	Base-A	buffer 1
23	1211	T13O	Robô 2	Movimentação	Base-A	Saída
24	1211	T15O	Robô 1	Movimentação	Tray-A	Saída
25	1212	T15O	Robô 1	Movimentação	Tray-A	Saída
26	1411	T112	Robô 2	Movimentação	Base-A	buffer 2
27	1412	T12	Estação 2	Operação	Base-A	buffer 2
28	1712	T123	Robô 2	Movimentação	Base-A	buffer 3
29	1713	T145	Robô 1	Movimentação	Tray-A	buffer 5
30	1714	T1I5	Robô 1	Movimentação	Tray-A	buffer 5
31	1715	T13	Estação 3	Operação	Base-A	buffer 3
32	2015	T13O	Robô 2	Movimentação	Base-A	Saída
33	2015	T15O	Robô 1	Movimentação	Tray-A	Saída
34	2016	T15O	Robô 1	Movimentação	Tray-A	Saída

Simulação 6

Entrada: dois produtos tipo-A e dois produtos tipo-B

Tabela 11: Saída da simulação 6

Step	u.t	Tansição Disparada	Recurso	Evento	Peça	Destino
1	1	T2I3	Robô 2	Movimentação	Base-B	buffer 3
2	2	T1I0	Robô 2	Movimentação	Base-A	buffer 0
3	2	T2I5	Robô 1	Movimentação	Tray-B	buffer 5
4	3	T23	Estação 3	Operação	Base-B	buffer 3
5	3	T1I4	Robô 1	Movimentação	Tray-A	buffer 4
6	4	T11	Estação 1	Operação	Base-A	buffer 1
7	303	T232	Robô 2	Movimentação	Base-B	buffer 2
8	304	T2I4	Robô 1	Movimentação	Tray-B	buffer 4
9	305	T22	Estação 2	Operação	Base-B	buffer 2
10	605	T220	Robô 2	Movimentação	Base-B	buffer 0
11	606	T112	Robô 2	Movimentação	Base-A	buffer 2
12	607	T12	Estação 2	Operação	Base-A	buffer 2
13	607	T21	Estação 1	Operação	Base-B	buffer 1
14	907	T123	Robô 2	Movimentação	Base-A	buffer 3
15	908	T1I5	Robô 1	Movimentação	Tray-A	buffer 5
16	909	T145	Robô 1	Movimentação	Tray-A	buffer 5
17	910	T13	Estação 3	Operação	Base-A	buffer 3
18	1007	T21O	Robô 2	Movimentação	Base-B	Saída
19	1007	T24O	Robô 1	Movimentação	Tray-B	Saída
20	1008	T1I0	Robô 2	Movimentação	Base-A	buffer 0
21	1009	T1I4	Robô 1	Movimentação	Tray-A	buffer 4
22	1010	T11	Estação 1	Operação	Base-A	buffer 1
23	1210	T13O	Robô 2	Movimentação	Base-A	Saída
24	1210	T15O	Robô 1	Movimentação	Tray-A	Saída
25	1211	T15O	Robô 1	Movimentação	Tray-A	Saída
26	1211	T2I3	Robô 2	Movimentação	Base-B	buffer 3
27	1212	T2I5	Robô 1	Movimentação	Tray-B	buffer 5
28	1213	T23	Estação 3	Operação	Base-B	buffer 3
29	1513	T232	Robô 2	Movimentação	Base-B	buffer 2
30	1514	T2I4	Robô 1	Movimentação	Tray-B	buffer 4
31	1515	T22	Estação 2	Operação	Base-B	buffer 2
32	1815	T220	Robô 2	Movimentação	Base-B	buffer 0
33	1816	T112	Robô 2	Movimentação	Base-A	buffer 2
34	1817	T21	Estação 1	Operação	Base-B	buffer 1
35	1817	T12	Estação 2	Operação	Base-A	buffer 2
36	2117	T123	Robô 2	Movimentação	Base-A	buffer 3
37	2118	T145	Robô 1	Movimentação	Tray-A	buffer 5
38	2119	T1I5	Robô 1	Movimentação	Tray-A	buffer 5
39	2120	T13	Estação 3	Operação	Base-A	buffer 3

Step	u.t	Tansição Disparada	Recurso	Evento	Peça	Destino
40	2217	T21O	Robô 2	Movimentação	Base-B	Saída
41	2217	T24O	Robô 1	Movimentação	Tray-B	Saída
42	2420	T13O	Robô 2	Movimentação	Base-A	Saída
43	2420	T15O	Robô 1	Movimentação	Tray-A	Saída
44	2421	T15O	Robô 1	Movimentação	Tray-A	Saída

Os resultados mostram que a técnica aplicada evita os eventos que levam o sistema a um estado que dá condição para a ocorrência de *deadlock*. Além disso, ele permite paralelismo e sincronismo dos recursos, isto é, o sistema utilizou o máximo de recursos possíveis dentro das condições em que o sistema não entre em estado de *deadlock*.

5.4 Considerações finais

Além do cenário apresentado por Wu e Zhou (2008), a proposta foi implementada em outros cenários como o proposto por Piroddi (2007) e Fanti (2004). Em todos os cenários o método proposto gerou um sistema livre de *deadlock* dentro do escopo abordado.

Considerando as definições estudadas sobre modelos livres de *deadlock*, é possível afirmar que a estratégia de resolução proposta é válida dentro do contexto abordado. A partir destes resultados pode-se afirmar que a estratégia é aplicável em Sistemas de Manufatura Automatizados similares.

Capítulo 6

CONCLUSÕES

Foram estudadas várias abordagens voltadas para a resolução de *deadlock* em SMA, cada estratégia pode trabalhar em níveis diferenciados do SMA (projeto, planejamento, programação e controle). Porém, o foco da pesquisa foi na resolução do problema de *deadlock* nos estágios de programação e controle.

As abordagens que usam estratégias de resolução *Deadlock Prevention* usando modelos de rede de Petri encontram um desafio em relação a complexidade da rede gerada, ou seja, quanto mais complexo o SMA, maior será o modelo gerado.

As abordagens que usam modelos redes de Petri aplicando políticas de controle *Deadlock Avoidance*, se deparam com o desafio da permissividade.

Os modelos de rede de Petri possuem atributos que podem representar grande parte das características de um sistema de manufatura automatizado. A abordagem modular, encontrada nas redes de Petri Coloridas, entre outras, tem como objetivo tratar partes diferentes do sistema de manufatura, facilitando modelagem e representação do sistema. Dentre as técnicas abordadas redes de Petri são modelos sistemáticos, modulares para análises, simulação, planejamento e controle de FMSs.

O objetivo desse trabalho foi propor um método de resolução de *deadlocks* no controle de sistemas de manufatura automatizados utilizando as redes de Petri Coloridas. Esse método aborda as estratégias de resolução de *deadlock* existentes para o projeto e implementação do controle da produção, em Sistemas de Manufatura Automatizados.

Foi proposta uma forma de modelagem que favorecesse a aplicação do método, tratando os elementos de forma modular e orientada aos recursos do sistema.

O método proposto possui características das estratégias de *deadlock prevention* devido ao fato da modelagem fornecer elementos adicionais que podem auxiliar na resolução dos *deadlocks* e características da estratégia de *deadlock avoidance* devido a

possibilidade de o modelo ser utilizado para o controle do SMA e utilizar informações em tempo real para induzir o sistema a funcionar sem a ocorrência dos *deadlocks*.

Observando os resultados das simulações, pode-se verificar que em nenhum momento o sistema entrou em estado de *deadlock*. Ou seja, todas as peças foram montadas e as transições que levam as marcações para os *places* finais foram disparadas.

Também foi observado que o modelo do controlador permitiu um número máximo de peças em operação simultaneamente no sistema de forma que não ocorressem condições para a ocorrência de *deadlock*.

Considerando as definições estudadas sobre modelos livres de *deadlock*, é possível afirmar que a estratégia de resolução proposta é válida dentro do contexto abordado.

6.1 Trabalhos Futuros

A partir dos resultados atingidos pela proposta, alguns trabalhos futuros podem ser relacionados:

- Aprimorar o método para encontrar as regiões críticas;
- Propor a técnica de resolução de *deadlock* em Sistemas de Alocação de Recursos do tipo ST-RAS, AND-RAS e AND/OR-RAS;
- Tratar as informações incorporadas nos fluxos adicionais das marcações utilizando técnicas de Inteligência Artificial (I.A.);

REFERÊNCIAS

AGUIRRE, L. A. Enciclopédia de Automática Controle e Automação. 1 ed. São Paulo: Blucker, 2007. 450 p. (Teoria de controle - Enciclopédias).

BHATTACHARYA, R.; BANDYOPADHYAY, S. An improved strategy to solve shop deadlock problem based on the study on existing benchmark recovery strategies.**International Journal of Advanced Manufacturing Technology**, v. 47, n. 1-4, p. 351-364, 2010.

BICHO, A. L. **Controle de Animações por Computador utilizando redes de Petri**. 2001. f. Dissertação (Mestrado) - Departamento de Engenharia Elétrica, Universidade Estadual de Campinas, Campinas.

CHEW, S. F.; LAWLEY, M. A. Robust supervisory control for production systems with multiple resource failures.**IEEE Transactions on Automation Science and Engineering**, IEEE, v. 3, n. 3, p. 309-323, 2006. Disponível em: http://dx.doi.org/10.1109/TASE.2005.861397. Acesso em: 04/05/2010.

COFFMAN, E. G.; ELPHICK, M.; SHOSHANI, A. System Deadlocks.**ACM Comput. Surv.**, v. 3, n. 2, p. 67-78, 1971.

DESROCHERS, A. A.; AL-JAAR, R. Y. Application of Petri nets in manufacturing systems. 1 ed. New York: IEEE Press, 1995. 348 p. (Manufacturing).

DOTOLI, M.; FANTI, M. P. Deadlock detection and avoidance strategies for automated storage and retrieval systems.**IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews**, Dipartimento di Elettrotecnica ed Elettronica, Politecnico di Bari, 70125 Bari, Italy, v. 37, n. 4, p. 541-552, 2007. Disponível em: http://dx.doi.org/10.1109/TSMCC.2007.897690. Acesso em: 04/05/2010.

D'SOUZA, K. A.; KHATOR, S. K. System reconfiguration to avoid deadlocks in automated manufacturing systems.**Computers and Industrial Engineering**, Hampton Univ, Hampton, United States, v. 32, n. 2, p. 455-465, 1997. Disponível em: http://dx.doi.org/10.1016/S0360-8352(96)00216-1. Acesso em: 04/05/2010.

ELMEKKAWY, T. Y.; ELMARAGHY, H. A. Real-time scheduling with deadlock avoidance in flexible manufacturing systems.**International Journal of Advanced Manufacturing Technology**, Intelligent Mfg. Syst. (IMS) Center, University of Windsor, Windsor, Ont. N9B 3P4, Canada, v. 22, n. 3-4, p. 259-270, 2003. Disponível em: http://dx.doi.org/10.1007/s00170-002-1468-y. Acesso em: 04/05/2010.

EZPELETA, J.; COLOM, J. M.; MARTINEZ, J. Petri net based deadlock prevention policy for flexible manufacturing systems.**IEEE Transactions on Robotics and Automation**, Universidad de Zaragoza, Zaragoza, Spain, v. 11, n. 2, p. 173-184, 1995. Disponível em: http://dx.doi.org/10.1109/70.370500. Acesso em: 04/05/2010.

EZPELETA, J.; TRICAS, F.; GARCIA-VALLES, F.; COLOM, J. M. A banker's solution for deadlock avoidance in FMS with flexible routing and multiresource states.**IEEE Transactions on Robotics and Automation**, Depto. Info. Ingenieria Sistemas, Centro Politecnico Superior, Universidad de Zaragoza, 50015 Zaragoza, Spain, v. 18, n. 4, p. 621-625, 2002. Disponível em: http://dx.doi.org/10.1109/TRA.2002.801048. Acesso em: 04/05/2010.

EZPELETA, J.; VALK, R. A polynomial deadlock avoidance method for a class of nonsequential resource allocation systems.**IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans**, Department of Computer Science, Aragon Institute for Engineering Research (I3A), University of Zaragoza, 50018 Zaragoza, Spain, v. 36, n. 6, p. 1234-1243, 2006. Disponível em: http://dx.doi.org/10.1109/TSMCA.2006.878963. Acesso em: 04/05/2010.

FANTI, M. P. Deadlock Resolution Strategy for Automated Manufacturing Systems Including Conjunctive Resource Service.**IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.**, Electrical Engineering Department, Polytechnic of Bari, Bari, Italy, v. 34, n. 1, p. 80-92, 2004. Disponível em: http://dx.doi.org/10.1109/TSMCA.2003.822377. Acesso em: 04/05/2010.

FANTI, M. P.; MAIONE, B.; TURCHIANO, B. Comparing digraph and Petri net approaches to deadlock avoidance in FMS.**IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics**, IEEE, v. 30, n. 5, p. 783-798, 2000. Disponível em: http://dx.doi.org/10.1109/3477.875452. Acesso em: 04/05/2010.

FANTI, M. P.; ZHOU, M. Petri net approaches to deadlock modeling and resolution in automated manufacturing. In: 3, 2002, Yasmine Hammamet, Tunisia. **Proceedings.** Institute of Electrical and Electronics Engineers Inc., p. 142-147. Disponível em: http://dx.doi.org/10.1109/ICSMC.2002.1176026. Acesso em: 04/05/2010.

GROOVER, M. P. Fundamentals of modern manufacturing. 3 ed. Hoboken: John Wiley & Sons, 2007. 1022 p. (Industrial Engineering).

GUREL, A.; BOGDAN, S.; LEWIS, F. L. Matrix approach to deadlock-free dispatching in multi-class finite buffer flowlines.**IEEE Transactions on Automatic Control**, Department of Electrical and Electronic Engineering, Eastern Mediterranean University, via Mersin 10, Famagusta, Turkey, v. 45, n. 11, p. 2086-2090, 2000. Disponível em: http://dx.doi.org/10.1109/9.887631. Acesso em: 04/05/2010.

HOLT, R. C. Some deadlock properties of computer systems.**Computing Surveys**, v. 4, n. 1, p. 179-196, 1972.

HSIEH, F.-S. Reconfigurable fault tolerant deadlock avoidance controller synthesis for assembly production processes. In: 4, 2000, Nashville, TN, USA. **Proceedings.** IEEE, p. 3045-3050.

HSIEH, F.-S. Deadlock avoidance control for assembly processes with flexible routing and unreliable machines. In: 4, 2003, Washington, DC, United states. **Proceedings.** Institute of Electrical and Electronics Engineers Inc., p. 3396-3401. Disponível em: http://dx.doi.org/10.1109/ICSMC.2003.1244414. Acesso em: 04/05/2010.

HSIEH, F.-S. Fault-Tolerant Deadlock Avoidance Algorithm for Assembly Processes.**IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans**, v. 34, n. 1, p. 65-79, 2004.

HUANG, Y.-S. Design of deadlock prevention supervisors using Petri nets.**International Journal of Advanced Manufacturing Technology**, Department of Aeronautical Engineering, Chung Cheng Institute of Technology, National Defense University, 335 Taoyuan, Taiwan, v. 35, n. 3-4, p. 349-362, 2007. Disponível em: http://dx.doi.org/10.1007/s00170-006-0708-y. Acesso em: 04/05/2010.

INA. Integrated Net Analizer, a software tool for analysis of Petri nets, Version 2.2, 2003. Disponível em: http://www2.informatik.hu-berlin.de/~starke/ina.html. Acesso em: 28/05/2010.

JENSEN, F. V.; CHRISTENSEN, H. I.; NIELSEN, J. Bayesian methods for interpretation and control in multiagent vision systems. In: 1708, 1992, Orlando, FL, USA. **Proceedings.** Publ by Int Soc for Optical Engineering, p. 536-548.

JENSEN, K.; KRISTENSEN, L. M. Coloured Petri Nets. 1 ed. New York: Springer, 2009. 381 p. (ACM Computing).

LI, Z.; SHPITALNI, M. Smart deadlock prevention policy for flexible manufacturing systems using Petri nets.**IET Control Theory and Applications**, School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China, v. 3, n. 3, p. 362-374, 2009. Disponível em: http://dx.doi.org/10.1049/iet-cta:20070399. Acesso em: 04/05/2010.

LI, Z.; UZAM, M.; ZHOU, M. Deadlock control of concurrent manufacturing processes sharing finite resources.**International Journal of Advanced Manufacturing Technology**, School of Electro-Mechanical Engineering, Xidian University, No.2 South Taibai Road, Xi'an 710071, China, v. 38, n. 7-8, p. 787-800, 2008. Disponível em: http://dx.doi.org/10.1007/s00170-007-1125-6. Acesso em: 18/05/2010.

MAGGIO, E. G. R. **Uma heurística para a programação da produção de sistemas flexíveis de manufatura usando modelagem em redes de Petri**.107f. Dissertação (Mestrado) - Departamento de Ciência da Computação, Universidade Federal de São Carlos, São Carlos, 2005.

MAIONE, G. D., F. A Petri net and digraph-theoretic approach for deadlock avoidance in flexible manufacturing systems. In: Systems, Man, and Cybernetics, 1998. IEEE, 1, p. 605-610.

MOHAN, S.; YALCIN, A.; KHATOR, S. Controller design and performance evaluation for deadlock avoidance in automated flexible manufacturing cells. In: 20, 2004, **Proceedings.** Elsevier Ltd, p. 541-551. Disponível em: http://dx.doi.org/10.1016/j.rcim.2004.07.004. Acesso em: 04/05/2010.

MONTEVECHI, J. A. B.; JUNIOR, O. M.; MIYAGI, P. E. Sistemas de Manufatura. In: **Enciclopédia de Automática Controle & Automação**. São Paulo: Blucker, 2007. p. 247-287.

MORANDIN JUNIOR, O.; KATO, E. R. R. Virtual Petri nets as a modular modeling method for planning and control tasks of FMS. In: 2, 2003, Washington, DC, United states. **Proceedings.** Institute of Electrical and Electronics Engineers Inc., p. 1521-1527. Disponível em: http://dx.doi.org/10.1109/ICSMC.2003.1244627. Acesso em: 04/05/2010.

MURATA, T. Petri nets: properties, analysis and applications.**Proceedings of the IEEE**, Univ of Illinois, Chicago, IL, USA, v. 77, n. 4, p. 541-580, 1989. Disponível em: http://dx.doi.org/10.1109/5.24143. Acesso em: 04/05/2010.

OLIVEIRA JUNIOR, M. N. **Estimativa do consumo de energia devido ao software: uma abordagem baseada em redes de Petri coloridas**. 217 f. Tese (Doutorado) - Centro de Informática, Universidade Federal de Pernambuco, Recife, 2006.

PERKUSICH, A.; LIMA, A. M. N. Redes de Petri. In: **Enciclopédia de Automática Controle & Automação**. São Paulo: Blucher, 2007. p. 313-332.

PIRODDI, L.; CORDONE, R.; FUMAGALLI, I. Selective siphon control for deadlock prevention in Petri nets.**IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans**, Dipartimento di Elettronica e Informazione, Politecnico di Milano, 20133 Milano, Italy, v. 38, n. 6, p. 1337-1348, 2008. Disponível em: http://dx.doi.org/10.1109/TSMCA.2008.2003535. Acesso em: 04/05/2010.

PIRODDI, L.; CORDONE, R.; FUMAGALLI, I. Combined siphon and marking generation for deadlock prevention in Petri nets.**IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans**, Dipartimento di Elettronica e Informazione, Politecnico di Milano, 20133 Milano, Italy, v. 39, n. 3, p. 650-661, 2009. Disponível em: http://dx.doi.org/10.1109/TSMCA.2009.2013189. Acesso em: 04/05/2010.

PIRODDI, L.; CORDONE, R.; FUMAGALLI, I. Efficient deadlock prevention in petri nets through the generation of selected siphons. In: 2009, St. Louis, MO, United states. **Proceedings.** Institute of Electrical and Electronics Engineers Inc., p. 5006-5011. Disponível em: http://dx.doi.org/10.1109/ACC.2009.5159861. Acesso em: 04/05/2010.

PIRODDI, L.; COSSALTER, M.; FERRARINI, L. A resource decoupling approach for deadlock prevention in FMS.**International Journal of Advanced Manufacturing Technology**, Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milan 20133, Italy, v. 40, n. 1-2, p. 157-170, 2009. Disponível em: http://dx.doi.org/10.1007/s00170-007-1319-y. Acesso em: 04/05/2010.

REVELIOTIS, S. A. Accommodating FMS operational contingencies through routing flexibility.**IEEE Transactions on Robotics and Automation**, Georgia Inst of Technology, Atlanta, United States, v. 15, n. 1, p. 3-19, 1999. Disponível em: http://dx.doi.org/10.1109/70.744598. Acesso em: 04/05/2010.

ROSZKOWSKA, E. Supervisory Control for Deadlock Avoidance in Compound Processes.**IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans**, v. 34, n. 1, p. 52-64, 2004.

TANENBAUM, A. S. Modern Operating Systems. 2 ed. Englewood Cliffs: Prentice-Hall, 1992. 786 p. (Computers).

TRICAS, F.; GARCIA-VALLES, F.; COLOM, J. M.; EZPELETA, J. A Petri net structure-based deadlock prevention solution for sequential resource allocation systems. In: 2005, 2005, Barcelona, Spain. **Proceedings.** Institute of Electrical and Electronics Engineers Inc., p. 271-277. Disponível em: http://dx.doi.org/10.1109/ROBOT.2005.1570131. Acesso em: 04/05/2010.

UZAM, M. An optimal deadlock prevention policy for flexible manufacturing systems using Petri net models with resources and the theory of regions.**International Journal of Advanced Manufacturing Technology**, Nigde Universitesi, Muhendislik-Mimarlik Fakultesi, Elektrik-Elektronik Muhendisligi Bolumu, Nigde, Turkey, v. 19, n. 3, p. 192-208, 2002. Disponível em: Acesso em: 04/05/2010.

UZAM, M. The use of the Petri net reduction approach for an optimal deadlock prevention policy for flexible manufacturing systems.**International Journal of Advanced Manufacturing Technology**, Muhendislik-Mimarlik Fakultesi, Elek.-Elektron. Muhendisligi Bolumu, Nigde Universitesi, Kampus, 51100 Nigde, Turkey, v. 23, n. 3-4, p. 204-219, 2004. Disponível em: http://dx.doi.org/10.1007/s00170-002-1526-5. Acesso em: 04/05/2010.

UZAM, M.; LI, Z.; ZHOU, M. Identification and elimination of redundant control places in petri net based liveness enforcing supervisors of FMS.**International Journal of Advanced Manufacturing Technology**, Muhendislik-Mimarlik Fakultesi, Elektrik-Elektronik Muhendisligi Bolumu, Kampus, Nigde Universitesi, Nigde 51200, Turkey, v. 35, n. 1-2, p. 150-168, 2007. Disponível em: http://dx.doi.org/10.1007/s00170-006-0701-5. Acesso em: 18/05/2010.

UZAM, M.; ZHOU, M. An iterative synthesis approach to Petri net-based deadlock prevention policy for flexible manufacturing systems.**IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans**, Nigde Universitesi, Muhendislik-Mimarlik Fakultesi, Elektrik-Elektronik Muhendisligi Bolumu, Kampus, 51200 Nigde, Turkey, v. 37, n. 3, p. 362-371, 2007. Disponível em: http://dx.doi.org/10.1109/TSMCA.2007.893484. Acesso em: 18/05/2010.

WU, N. Avoiding deadlocks in automated manufacturing systems with shared material handling system. In: Proceedings - IEEE International Conference on Robotics and Automation, 3, 1997, Albuquerque, NM, USA. IEEE, Piscataway, NJ, United States, p. 2427-2432

WU, N.; ZHOU, M. Deadlock resolution in automated manufacturing systems with robots.**IEEE Transactions on Automation Science and Engineering**, Department of Mechatronics Engineering, Guangdong University of Technology, Guangzhou 510090, China, v. 4, n. 3, p. 474-480, 2007. Disponível em: http://dx.doi.org/10.1109/TASE.2006.888049. Acesso em: 04/05/2010.

WU, N.; Zhou, M ; LI, Z. Resource-oriented Petri net for deadlock avoidance in flexible assembly systems.**IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans**, v. 38, n. 1, p. 56-69, 2008.

ZANDONG, H.; LEE, G. Application of Petri nets for deadlock analysis and avoidance in flexible manufacturing systems.**International Journal of Advanced Manufacturing Technology**, Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China, v. 25, n. 7-8, p. 735-742, 2005. Disponível em: http://dx.doi.org/10.1007/s00170-003-1907-4. Acesso em: 04/05/2010.

ZHOU, M.; DICESARE, F. Petri Net Syntesis for Discrete Event Control of Manufacturing Systems. 1 ed. Norwell, Massachusetts: Kluwer Academic Publishers, 1993. p. (The Kluwer international series in engineering and computer science.).

ZHOU, M.; FANTI, M. P. Deadlock Resolution in Computer-Integrated Systems. 1 ed. New Jersey: CRC Press, 2004. 696 p. (Computer Engineering).

APÊNDICE A

CPN TOOLS

6.2 Considerações Iniciais

Neste apêndice será apresentado como criar e editar redes de Petri utilizando a ferramenta CPN Tools.

Mesmo que a CPN Tools foi desenvolvida especialmente para a criação de redes de Petri Coloridas, ela possui suporte para outras variações e extensões de rede de Petri.

Durante a criação deste tutorial a ferramenta estava na sua versão 3.2.2. Apesar de existirem versões para Mac OS X e Linux, neste tutorial será apresentada a ferramenta para Windows, por tanto, somente algumas partes podem não ser compreendidas caso queira seguir estas instruções em outro sistema operacional, como a instalação ou alguma parte onde será necessário salvar ou abrir algum arquivo.

6.3 Download e instalação

Para fazer o download da última versão é preciso acessar o site dos desenvolvedores da ferramenta (http://cpntools.org/download). Após o baixar o instalador, execute-o.

No início da instalação irá aparecer a janela de apresentação como é mostrado na Figura 59. Clique no botão “Next” para passar para as próximas configurações de instalação.

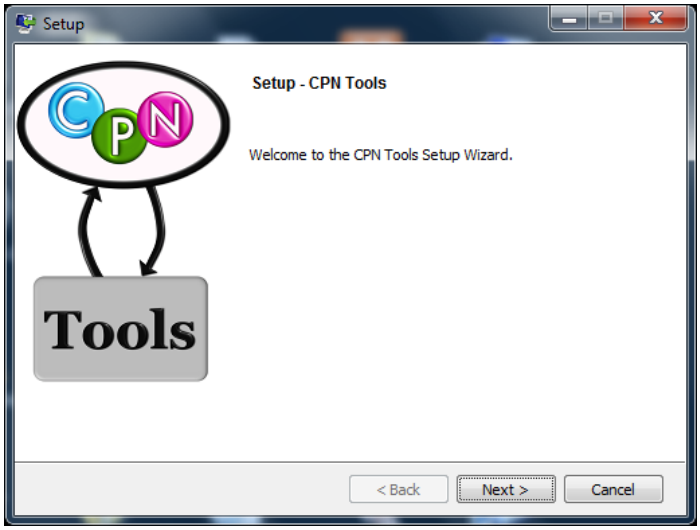


Figura 62 : Janela de apresentação do instalador do CPN Tools 3.2.2.

Na tela seguinte, será requisitado o diretório onde a ferramenta será instalada, no campo já vem preenchido um diretório padrão, caso queira alterar, o diretório escolhido pode ser digitado no campo ou é possível escolher através de uma lista em árvores das pastas existentes no computador clicando no botão com o desenho de uma pasta amarela do lado direito do campo, como é apresentado na Figura 60. Após a escolha do diretório clique novamente em "*Next"*.

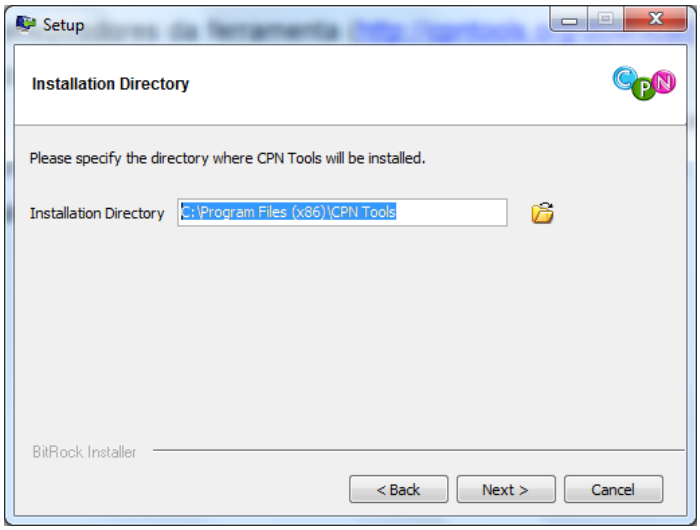


Figura 63: Janela de configuração do diretório para a instalação.

Na próxima tela, como é mostrado na Figura 61, são apresentados os componentes opcionais para a instalação. Quando cada um deles são clicados, ao lado direito da janela são apresentadas suas funcionalidades. Em caso de dúvidas deixe a seleção que vem por padrão. Clicando novamente em "*Next*" o assistente de instalação estará pronto para iniciar a instalação de acordo com as configurações escolhidas. Caso queira alterar algumas dessas opções, é só clicar nos botões "*Back"* para retroceder entre as telas apresentadas.

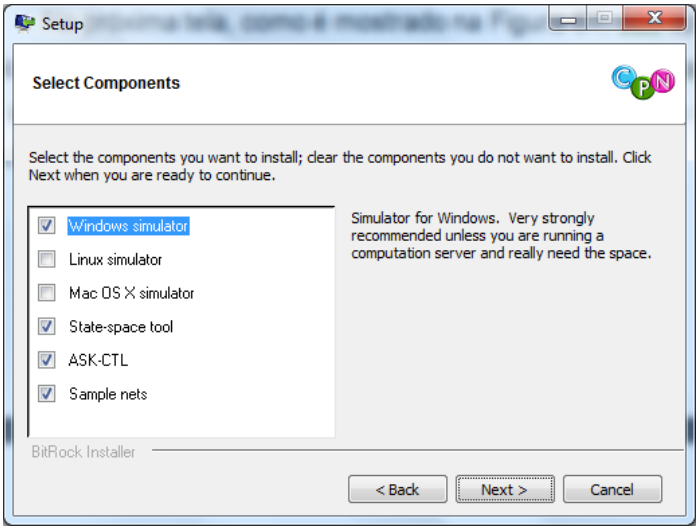


Figura 64: Tela de seleção de componentes opcionais.

Assim que todas as opções foram escolhidas, na tela "*Ready to Install"*, dê mais um clique no botão "*Next"*. Então o assistente irá iniciar o processo de instalação. Unado o processo terminar com sucesso, irá aparecer a tela de conclusão, dizendo que a instalação foi um sucesso. Agora é só clicar no botão "*Finish*" para sair do instalador.

O Ícone de atalho da ferramenta será instalado no menu iniciar do Windows, para abri-lo só é preciso dar um clique duplo no novo ícone instalado onde está escrito "*CPN Tools*".

6.4 Interface gráfica

A seguir, serão apresentados os principais componentes que facilitam a utilização do CPN Tools. Eles se resumem em janelas

6.4.1 Index

Index é uma parte reservada ao lado esquerdo da janela do CPN Tools que contém listadas algumas ferramentas úteis para edição e configuração das redes de Petri. Nele também estão as Paletas (*Tool Boxes*) que serão detalhadas a seguir.

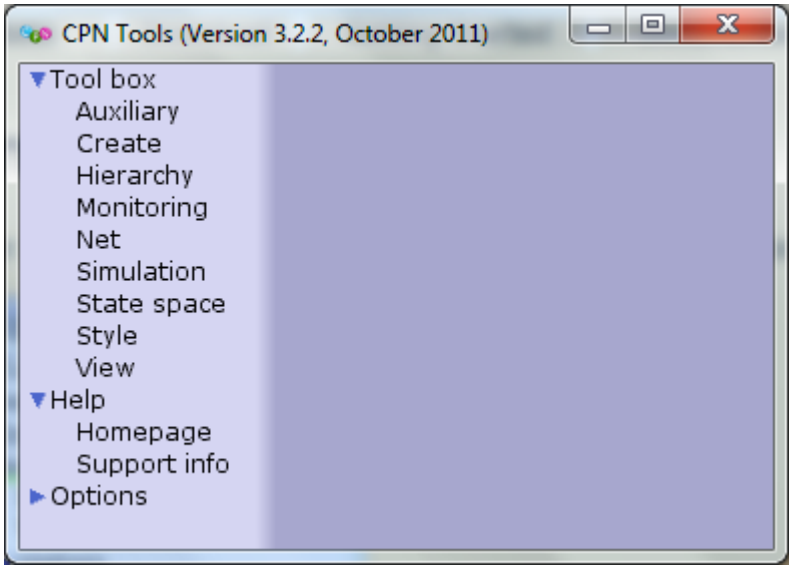
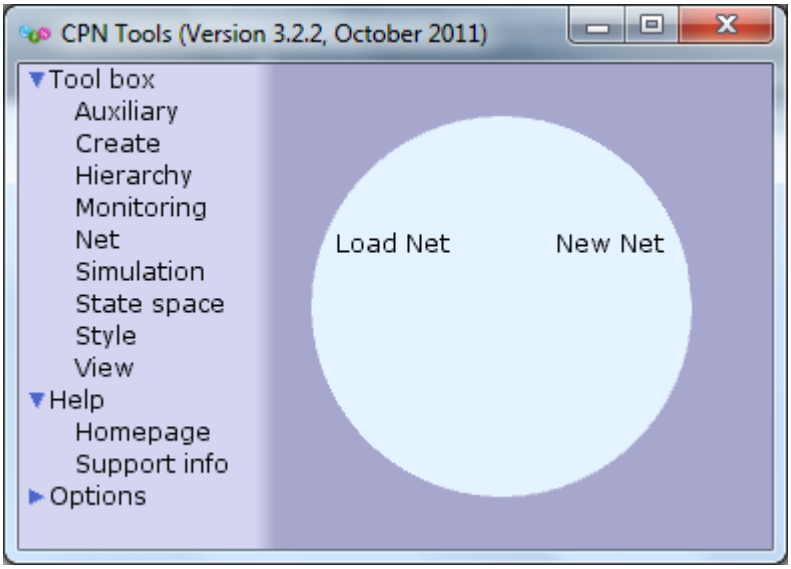


Figura 65: Janela inicial do CPN Tools

6.4.2 Menus de Contexto

Os menus de contextos são muito utilizados no CPN Tools, inclusive, algumas ações são possíveis de serem executadas somente pelos menus de contexto. A vantagem disto é que essa ações acabam ficando simples de serem encontradas, em contrapartida algumas ações que poderiam ser executadas por atalhos no teclado, como a exclusão de um objeto ou desfazer a última alteração, não são permitidas.

Quando a tela inicial do CPN Tolls é clicada com o botão direto, o menu de contexto irá abrir com duas opções (ver na Figura 63), A opção "*Load Net*" serve para abrir um arquivo salvo de um modelo de rede de Petri já construído anteriormente.



A opção “*New Net*” abre um quadro (*Binder*) para a visualização, criação e edição dos modelos. Após a abertura de um quadro, tanto novo como salvo, é necessária a abertura de algumas Paletas para que seja possível criar, editar e simular e analisar as redes criadas.

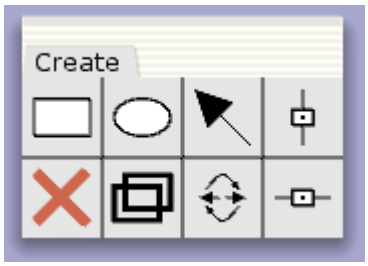
6.4.3 Paletas

A seguir serão apresentadas as principais paletas úteis para criar um modelo de rede de Petri. Com essas ferramentas é possível criar algumas das variações e extensões da rede de Petri existentes.

Todos os botões das paletas transformam o cursor do mouse para que este possa executar um tipo específico de ação no modelo de rede de Petri a ser editado. Essas ações serão explicadas a seguir.

É importante lembrar que o CPN Tools sempre irá salvar a última configuração de interface deixada pelo usuário antes que o aplicativo seja fechado, assim, quando o CPN Tools é executado em uma próxima vez, as paletas que foram deixadas abertas aparecerão no mesmo lugar da última sessão.

Criação



A paleta de criação apresentada na figura acima possui botões para a criação de novos elementos da rede de Petri. O botão com um retângulo é usado para criar novas transições, o botão com uma elipse é usado para criar novos *places*, quando este é acionado pelo mouse, o cursor toma a forma de um retângulo. Quando o quadro de edição é clicado, uma nova transição é criada no lugar em que foi feito o clique.

O mesmo mecanismo ocorre com o botão de criação de p*laces*, representado pelo botão com uma elipse. O botão com uma seta é utilizado para a criação de transições. Para que isto seja possível, é necessário que o modelo tenha ao menos uma transição e um *place* já criados, pelo fato de que um arco pode somente ligar uma transição com um *place*, independente de sua direção. Sendo assim, quando o botão é acionado, o cursor do mouse

se transforma em uma seta, então é só clicar em cima de uma transição e depois em cima de um *place*, o contrário também funcionará. A transição seguirá o sentido do último elemento clicado. Caso seja necessário trocar o sentido da transição depois que esta já foi clicada

Estilos

A paleta de estilo “Style” serve para facilitar a visualização da rede, dependendo do tipo de rede criado a paleta pode ser usada para diferenciar funções e setores da rede.

Para trocar o estilo (ou a cor) de um elemento, basta clicar em uma das cores da paleta e em seguida no elemento desejado, pode ser qualquer arco, transição, places, etc.

Visualização

Na paleta de visualização “View” é possível aproximar ou afastar o zoom do quadro de edição.

Simulação

A paleta de simulação é utilizada para visualizar o comportamento da rede de Petri modelada.

Clicando no primeiro botão e depois na janela de edição, a rede retornará a sua marcação inicial, fazendo o mesmo com o segundo botão, a simulação irá parar. Como terceiro botão é possível escolher manualmente qualquer transição habilitada para ser disparada. O quarto botão faz com que uma das transições habilitadas dipare

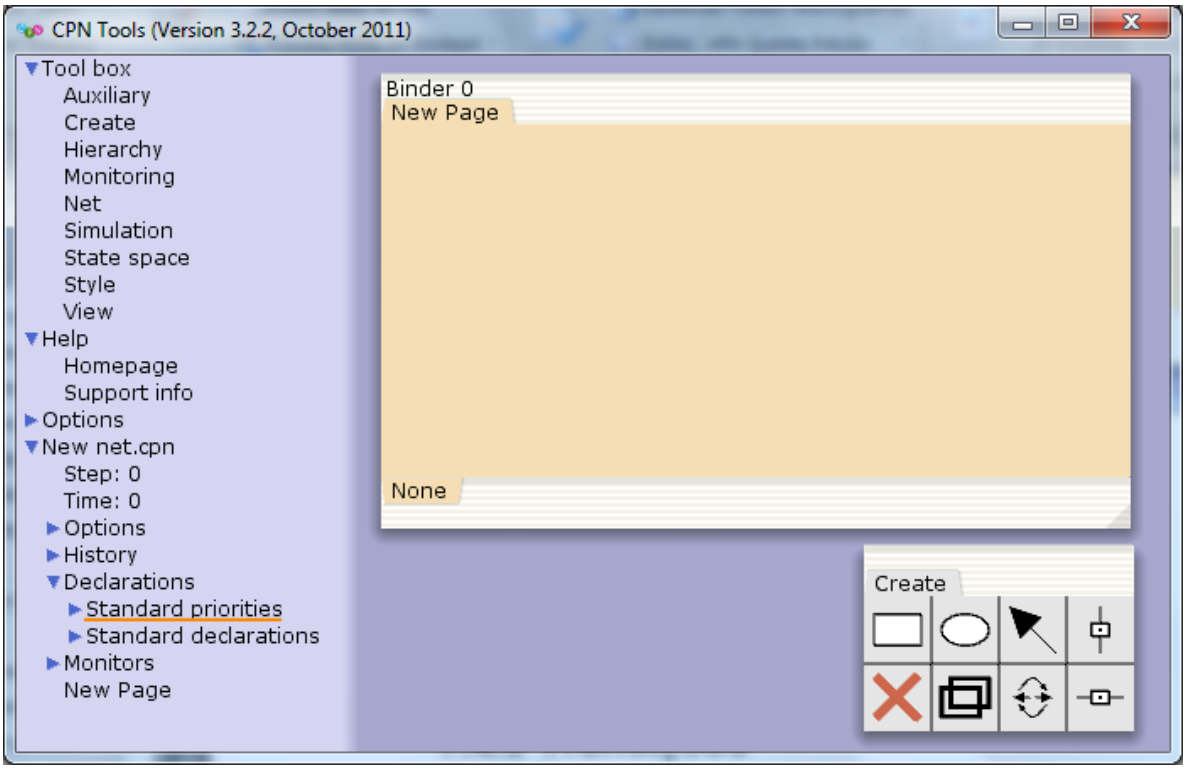
automaticamente (escolha arbitrária da própria ferramenta). Clicando no quinto botão, é possível escolher quantas transições consecutivas serão disparadas na simulação e o intervalo de tempo (em milissegundos) entre os disparos. Ao clicar no sexto botão é possível configurar quantos disparos pular, ou seja, depois de configurado, quando a janela de edição é clicada com este botão, a visualização mostrará a marcação após aquele número de disparos.

6.5 Criação e edição

A partir das ferramentas apresentadas na parte gráfica é possível criar e editar uma rede de Petri.

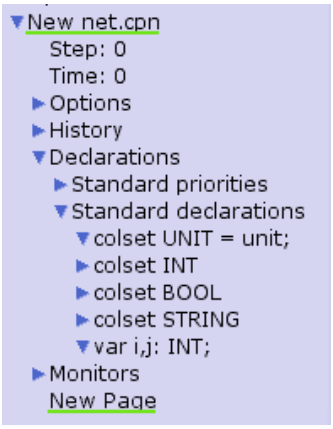
Para criar uma nova rede basta clicar em “Net” no “*Index*” e arrastar até a área de trabalho ao lado. Após fazer isso uma nova janela de edição irá aparecer e no “*Index*” irá aparecer um novo item chamado “*New net.cpn*”.

Abrindo o item “*New net.cpn*” com um clique no triângulo à esquerda do mesmo fará com que outros itens de configurações da rede apareça.



Dentro do item “*declarations*” contém uma chamado “Standard declarations” nela estão declaradas as cores padrões da rede de petri. UNIT é o tipo tradicional de marca em redes de petri, a cor é representada por marcações com o texto “unit”, ou seja, uma unidade.

INT é um tipo reservado para que as marcas dessa cor armazenem números inteiros. BOOL faz com que marcas desse tipo armazenem dados binários. STRING armazena informações em uma vetor de caracteres.



Além das cores padrões, outra cores podem ser criadas, para isso basta dar um clique no item "*Standard declarations*" e no menu contexto selecione "*New Declaration*", uma nova linha será criada, então basta editá-la começando com colset depois o nome, em seguida coloque o símbolo "=" e em seguida o conjunto de símbolos que será representado pelas marcas.

Além dos conjuntos de cores é possível declarar variáveis e rotinas. No exemplo da figura acima foram criadas duas variáveis.

6.6 Considerações finais

A ferramenta de modelagem de redes de Petri CPN Tool possui, além dessas, várias opções de modelagem e funcionalidades que podem ser vistas no site de documentação dos desenvolvedores acessando: http://cpntools.org/documentation/start.