UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS CORNÉLIO PROCÓPIO
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA
MESTRADO EM ENGENHARIA ELÉTRICA

CELSO MATSUMI KAWAMURA

CONTRIBUIÇÕES À ANÁLISE DE DESEMPENHO DE CÉLULAS DE MANUFATURA BASEADA NA TEORIA DE CONTROLE SUPERVISÓRIO

DISSERTAÇÃO

CORNÉLIO PROCÓPIO

2012

CELSO MATSUMI KAWAMURA

CONTRIBUIÇÕES À ANÁLISE DE DESEMPENHO DE CÉLULAS DE MANUFATURA BASEADA NA TEORIA DE CONTROLE SUPERVISÓRIO

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná como requisito parcial para obtenção do título de “Mestre em Engenharia Elétrica”.

Orientador: Prof. Dr. Marcos B. R. Vallim

Co-orientador: Prof. Dr. Rodrigo R. Sumar

CORNÉLIO PROCÓPIO

2012

K22c Kawamura, Celso Matsumi.

Contribuições à análise de desempenho de células de manufatura baseada na teoria de controle supervisório / Celso Matsumi Kawamura. – Cornélio Procópio : UTFPR, 2012.
97 f. : il. color. ; 28 cm.

Orientador: Prof. Dr. Marcos Banheti Rabello Vallim.
Co-orientador: Prof. Dr. Rodrigo Rodrigues Sumar.
Dissertação de Mestrado em Engenharia Elétrica – Universidade Tecnológica Federal do Paraná. Diretoria de Pesquisa e Pós-Graduação. Programa de Pós-Graduação em Engenharia Elétrica. Cornélio Procópio, 2012.
Referências: p. 91 - 94.

1. Sistemas de controle supervisório. 2. Controle de processo. 3. Sistemas de tempo discreto. I. Vallim, Marcos Banheti Rabello, orient. II. Sumar, Rodrigo Rodrigues, co-orient. III. Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Engenharia Elétrica. IV. Título.

CDD: 621.3

Ficha catalográfica desenvolvida pelo bibliotecário Romeu Righetti de Araujo – CRB/9-1676

Minstério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Cornélio Procópio
Diretoria de Pesquisa e Pós-Graduação
Programa de Pós-Graduação em Engenharia Elétrica
Mestrado em Engenharia Elétrica

TERMO DE APROVAÇÃO

CONTRIBUIÇÕES À ANÁLISE DE DESEMPENHO DE CÉLULAS DE MANUFATURA BASEADA NA TEORIA DE CONTROLE SUPERVISÓRIO

por

Celso Matsumi Kawamura

Esta Dissertação foi julgada adequada para obtenção do Título de "Mestre em Engenharia Elétrica" e aprovado em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná.

Cornélio Procópio, 28/06/2012.

Alessandro Goedtel, Doutor
Coordenador do Curso

Banca Examinadora:

Marcos B. R. Vallim, Prof. Dr.
Orientador

Rodrigo R. Sumar, Prof. Dr.
Co-orientador

Bruno Augusto Angélico, Doutor, UTFPR

Antonio E. Carrilho da Cunha, Doutor, IME

Dedico esse trabalho aos meus pais, por tudo que fizeram para proporcionar o prazer dessa conquista;

à minha irmã, Celly, quem primeiro me inspirou a seguir os caminhos exatos da engenharia;

à minha esposa, para mim sempre um exemplo de dedicação, comprometimento e responsabilidade;

e às minhas filhas Mariana e Nathalia, inúmeras vezes privadas da minha companhia, pela paciência e compreensão.

AGRADECIMENTOS

Nos últimos dois anos muitas pessoas fizeram parte dessa conquista, e talvez, ao mencionar apenas algumas, eu possa cometer injustiças ou omissão involuntária, mas assim mesmo ouso tentar.

Agradeço à Deus, que é por tudo e por todos.

À Elevadores Atlas Schindler S.A., nas pessoas do Sr. Jose Carlos A. Lusquiños, e Elias Gonçalves (*in memorian*) pela oportunidade e confiança no meu trabalho.

Ao meu amigo Alexandre Graeml, que foi quem me motivou e incentivou a dar os primeiros passos nos caminhos da pesquisa acadêmica.

Aos colegas e amigos do CIPECA, pelas horas compartilhadas de estudo, descontração, convivência e amizade.

Aos amigos do Departamento de Recursos da Elevadores Atlas Schindler S.A., pois foi com seu apoio que pude conciliar as atividades acadêmicas e profissionais.

À Márcia, da secretaria do CIPECA, sempre solícita, pela atenção oferecida aos alunos do Mestrado.

Ao Diego e Expedito, pelas valorosas contribuições gráficas.

Ao Edson Takigami, por me fazer perceber uma oportunidade.

Ao meu orientador Professor Marcos B. R. Vallim pela paciência, sabedoria e orientação pelas trilhas do conhecimento científico.

Ao corpo docente do Mestrado, por compartilhar conosco todo o conhecimento ao longo de nosso desenvolvimento.

Descobri como é bom chegar quando se tem paciência. E para se chegar, onde quer que seja, aprendi que não é preciso dominar a força, mas a razão. É preciso, antes de mais nada, querer.

Amyr Klink

RESUMO

KAWAMURA, Celso M. **CONTRIBUIÇÕES À ANÁLISE DE DESEMPENHO DE CÉLULAS DE MANUFATURA BASEADA NA TEORIA DE CONTROLE SUPERVISÓRIO.** 98 f. Dissertação – Mestrado em Engenharia Elétrica, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2012.

A crescente competitividade entre as indústrias de manufatura faz com que as empresas se preocupem cada vez mais com questões relacionadas à qualidade, produtividade e redução de custo. A concorrência internacional, bastante incisiva nos setores industriais de transformação de matérias-primas é, de fato, uma questão preocupante. Para tornarem-se competitivas, as empresas precisam extrair mais produtos com uma mesma quantidade de insumos e produzir mais, em menores frações de tempo, tornando as restrições de tempo e recursos cada vez mais severas. Melhores desempenhos e processos mais eficientes podem ser obtidos por meio da automação industrial. Porém, tais projetos são normalmente desenvolvidos sem a utilização de abordagens formais e os resultados obtidos dependem, quase que exclusivamente, da experiência dos projetistas. Não é possível afirmar que um conjunto de máquinas esteja operando sem qualquer situação de bloqueio, ou ainda, que esteja produzindo da maneira mais eficiente possível, tendo-se como critério o tempo de execução das operações. Por intermédio de abordagens formais, como a Teoria de Controle Supervisório (TCS), é possível a síntese automática de supervisores, que permitem a operação de uma planta, os quais, além de atender aos requisitos de segurança, evitam que o sistema atinja uma situação de bloqueio. Este trabalho apresenta algumas extensões dessa teoria, de modo que o supervisor ótimo satisfaça também às especificações de sequenciamento de operações e de prazo de finalização de tarefas, com a abordagem de autômatos temporizados. O problema de escalonamento de tarefas, diretamente relacionado ao planejamento de produção nas indústrias, é um problema de otimização, de tal maneira que a sequência de execução das etapas seja organizada para que o tempo total de processamento seja minimizado. Nessa pesquisa, uma proposta de um método para a obtenção de escalonamento de operações, com a utilização de autômatos temporizados e conceitos da teoria de controle supervisório, é apresentada. Propõe-se um algoritmo para a obtenção de uma sequência ótima de operações, tendo-se como critério o tempo total para a finalização do processo. O método é aplicado em uma célula automatizada em uma indústria fabricante de elevadores, onde foi possível determinar o tempo mínimo de processamento de um determinado item. Por meio da pesquisa de métodos e algoritmos para a obtenção de escalonamento de tarefas em ambientes fabris, procura-se atender às necessidades estratégicas das indústrias, na busca por sistemas de manufatura cada vez mais eficientes e competitivos.

Palavras-chave: sistemas a eventos discretos, controle supervisório, sistemas a eventos discretos temporizados, escalonamento na produção

ABSTRACT

KAWAMURA, Celso M. **CONTRIBUTIONS TO MANUFACTURING CELLS PERFORMANCE ANALYSIS BASED ON SUPERVISORY CONTROL THEORY.** 98 f. Dissertação – Mestrado em Engenharia Elétrica, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2012.

The growing competition among manufacturing industries causes companies to worry more and more with issues related to quality, productivity and cost reduction. International competition, quite incisive among raw material processing manufacturers, is actually a very concerning situation. To become more competitive, enterprises have to produce more with the same amount of raw materials, or in a shorter time, making the constraints of time and resources increasingly severe. A better performance and more efficient processes can be obtained by means of industrial automation. However, such projects are normally designed without any formal approach and the achieved results depend, more than anything else, on engineers' expertise. It cannot be stated that a machine set is working without any deadlock situation, or is producing in the most efficient way possible, considering operations´ execution time as efficiency criteria. By means of formal approaches such as Supervisory Control Theory (SCT), it is possible to synthesize supervisors automatically, that allow a plant to operate, in a way that, not only safety requirements are fulfilled, but the system is also prevented to reach a deadlock situation. This work presents this theory and its extensions, so the optimal supervisor satisfies task scheduling requirements and job completion deadlines, by using timed automata frameworks. The job shop scheduling problem, which directly relates to manufacturers´ production planning, is an optimization problem, where all the processing steps are organized in way that the total processing time is minimized. A methodology is presented for the job shop scheduling, by means of timed automata and supervisory control theory. An algorithm for acquiring an optimal operation sequence is proposed, considering the total time needed to finalize a process as the optimization criteria. The method is applied in an automated cell in a elevator manufacturing industry, where it was possible to obtain the minimal time needed to process a certain part. Through algorithms and methods for job shop scheduling in industrial environments, it is intended to meet the manufacturers´ strategic needs, in their search for manufacturing systems that are increasingly more efficient and competitive.

Keywords: discrete event systems, supervisory control, timed discrete event systems, job shop scheduling

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ADEF	Autômato Determinístico de Estado Finito.
CML	Controle Modular Local.
ME	Manufatura Enxuta.
MNSC	*marking nonblocking supervisory controller*.
MRP	*Manufacturing Resource Planning*.
OLX	*Linear Order Crossover*.
RSP	Representação por Sistema Produto.
SEDs	Sistemas a Eventos Discretos.
SEDTs	Sistemas a Eventos Discretos Temporizados.
TCS	Teoria de Controle Supervisório.

LISTA DE SÍMBOLOS

$u(t)$	vetor coluna de entrada do modelo de um sistema
$y(t)$	vetor coluna de saída do modelo de um sistema
Σ	alfabeto de eventos
L	linguagem
ε	cadeia vazia
Σ^*	fechamento *Kleene*
$\overline{L}$	prefixo fechamento de linguagem
L^*	fechamento Kleene de L
G	autômato
X	conjunto de estados de um autômato
f	função de transição entre estados de um autômato
x_0	estado inicial do autômato
X_m	conjunto de estados finais ou marcados
$\widehat{f}$	função estendida de f
$G_1 \| G_2$	composição síncrona entre G_1 e G_2
$L(G)$	linguagem gerada de G
$L_m(G)$	linguagem marcada de G
Σ_c	conjunto dos eventos controláveis
Γ	estrutura de controle para G
S	supervisor de controle
$supC(K)$	máxima linguagem controlável contida em K
A	conjunto de atividades
δ_{act}	função transição de atividades
a_0	atividade inicial
A_m	conjunto de atividades finais ou marcadas
σ	transição entre atividades
l_σ	limite de tempo inferior de uma atividade
u_σ	limite de tempo superior de uma atividade
Σ_{act}	conjunto de eventos temporizados
Σ_{spe}	conjunto de eventos prospectivos
Σ_{rem}	conjunto de eventos remotos
$\widehat{\delta}$	função estendida de δ
Σ_{hib}	conjunto de eventos proibitíveis
$Elig_G(s)$	subconjunto de eventos elegíveis da cadeia s em G
J_i	tarefa que consiste de uma sequência de operações
O_{im}	Operação da tarefa J_i que deve ser processada na máquina M_m
M_m	máquina m do conjunto M
μ	função de atribuição da operação p à máquina m
st	função que indica o tempo de início de cada operação
d	função que indica a duração de cada operação
p	operação

en	final de operação
D	tamanho do escalonamento
cl	peça do tipo coluna
fl	peça do tipo folha de porta
rf	peça do tipo reforço
Σ_{for}	conjunto dos eventos forçáveis
Σ_{unc}	conjunto dos eventos não controláveis

SUMÁRIO

1 INTRODUÇÃO

A necessidade das indústrias de manufatura em relação à melhoria contínua nos níveis de qualidade e produtividade, é atendida pelo desenvolvimento de células de trabalho parcial, ou até mesmo totalmente automatizadas, onde não há qualquer intervenção humana durante sua operação normal. Esse aumento de produtividade, por intermédio da gestão do tempo de manufatura como vantagem competitiva, é um conceito já consolidado. A maneira como as indústrias gerenciam o tempo na produção representa um dos fatores vitais para um desempenho ótimo de uma indústria frente a seus concorrentes (MONTGOMERY; PORTER, 1998). Outra vantagem relacionada com a produtividade diz respeito à gestão da manufatura, baseada nos conceitos da Manufatura Enxuta (ME) (MELO; SACOMANO, 2005). Nesse modelo, um dos elementos indispensáveis para a eficiência do sistema de produção puxada (*pullsystem*), e também para alcançar a desejada fluidez no processo enxuto, é a presença de um nível mínimo de automação (HARRIS; ROTHER, 2003). O aumento no nível de automação traz redução nos custos de mão-de-obra direta e propicia maior repetitividade nos processos de manufatura (SLACK, 1993).

Além de trazer benefícios do ponto de vista financeiro, as condições que eventualmente possam trazer prejuízos à saúde do operador podem ser evitadas com a utilização de máquinas e equipamentos automatizados.

Essa automação porém, é comumente feita sem as abordagens formais ou modelagens matemáticas encontradas na literatura, sendo os resultados obtidos baseados principalmente na experiência e inspiração do projetista ou programador responsável. Não se garante dessa maneira que as células de manufatura estejam operando segundo um desempenho ótimo em relação ao ordenamento na ocorrência dos eventos, ou então de modo seguro e livre de bloqueios.

Outra observação a ser feita é que, sendo o desenvolvimento feito basicamente de forma empírica, não se pode afirmar que o máximo desempenho das células tenha sido obtido.

A Teoria de Controle Supervisório (TCS) (RAMADGE; WONHAM, 1989) garante a execução síncrona e não bloqueante das atividades em uma célula de manufatura. Entretanto, somente um controle supervisório, não necessariamente implica em melhor desempenho da planta, considerando critérios de escalonamento ótimo de tarefas (WANG, 2004). Com a combinação das duas abordagens: i) obtenção de um controle minimamente restritivo e não bloqueante; e ii) sequenciamento ótimo aplicada a uma célula de manufatura; pode-se avaliar o desempenho, e melhorar as taxas de produção, ampliando a capacidade de controle da automação dos sistemas industriais.

1.1 ESCALONAMENTO DA PRODUÇÃO

Segundo a ANSI (*American National Standard Institute*), escalonamento é definido como “o planejamento com o qual uma fábrica produz um determinado número de peças, em determinado intervalo de tempo”, ou ainda, *scheduling* ou escalonamento, é o procedimento para se encontrar um método ideal para que determinado objetivo seja atingido, especificando a sequência ou o tempo máximo de execução de cada item ou processo (YAHYAOUI; FNAIECH, 2006).

O escalonamento de tarefas é um problema tipicamente encontrado na indústria de manufatura, onde uma das condições para se aumentar a competitividade reside na utilização de suas máquinas e equipamentos com a melhor eficiência possível, dependendo então, de um sequenciamento lógico de acesso a recursos e de determinadas restrições que devem ser impostas para o funcionamento automático dos sistemas.

Um sistema de manufatura é composto por um conjunto de máquinas, cada uma executando uma série de tarefas em diferentes conjuntos de peças e partes. Esses itens são processados de acordo com uma sequência pré-definida nas diversas máquinas que compõem a cadeia, sendo o transporte, ou a transferência das peças entre as células, feito por intermédio de robôs, esteiras, veículos auto-guiados, e em alguns casos, manualmente.

Um fluxo de processo é associado a cada peça que deve ser produzida, definindo a sequência de operações que deve ser executada na matéria prima, de modo que o produto acabado seja obtido. Cada máquina possui um tempo de processo, i.e., o tempo necessário para que esta execute as operações em cada item.

O objetivo do escalonamento na produção é, portanto, a obtenção de um sequenciamento ótimo de operações e atribuições das máquinas onde as operações devem ser executadas, incluindo nessa sequência os tempos de início e fim de tarefa (DHINGRA, 1992).

1.2 JUSTIFICATIVA E OBJETIVOS

Problemas de escalonamento na produção (*job shop scheduling*), são reconhecidamente encontrados em linhas produtivas de indústrias metalúrgicas, compostas de várias células de manufatura. Tais células normalmente contam com manipuladores automatizados, esteiras motorizadas, robôs e máquinas operatrizes onde o sequenciamento de tarefas deve ser feito de modo que se obtenha uma alocação ótima de recursos compartilhados (MENGCHU, 2007).

Essa busca por processos fabris mais eficientes, motiva a comunidade científica, com pesquisas sendo realizadas com o objetivo de se obter a máxima otimização no sequenciamento de operações em células e sistemas de manufatura (SILVA, 2011), (PINHA; QUEIROZ; CURY, 2010), (ABDEDDAIM; ASARIN; MALER, 2006) e (WALLACE; JENSEN; SOPARKAR, 1996).

Esses sistemas produtivos, conforme descritos, podem ser associados à classe de Sistemas a Eventos Discretos (SEDs), uma vez que a evolução de estados ocorre em pontos discretos no tempo,e é dirigida a eventos. Uma extensa literatura, com abordagens formais de desenvolvimento de lógicas de controle e modelagens desses sistemas, está disponível podendo ser citadas redes de Petri (KROGH; HOLLOWAY, 1991), redes de Petri controladas (MURATA, 1989) (HOLLOWAY; KROGH; GIUA, 1997), teoria das filas (KLEINROCK, 1975) e linguagens formais e autômatos (CASSANDRAS; LAFORTUNE, 2008).

Em qualquer linha produtiva, a ausência de uma coordenação lógica otimizada, pode ocasionar desvios na operação do conjunto, levando a situações não previstas na especificação de projeto, bloqueando por completo o processo. Células de produção, observadas isoladamente, possuem comportamentos que devem ser restringidos e supervisionados, de modo que executem a função especificada pelo sistema global. Isso é obtido com a introdução de um agente chamado *supervisor* (CURY, 2001), e cujo desenvolvimento é encontrado nos trabalhos de Ramadge e Wonham (RAMADGE; WONHAM, 1989).

1.2.1 OBJETIVO GERAL

Propor um método para a obtenção de um escalonamento ótimo de tarefas, com a utilização de modelos de autômatos temporizados e conceitos da teoria de controle supervisório.

A abordagem utilizada, dentro da TCS, será o conceito de controle supervisório modular de sistemas a eventos discretos temporizados, conforme proposto por Brandin e Wonham (1993), tendo como critério de avaliação para o escalonamento ótimo, o tempo mínimo necessário para a fabricação de determinado item em uma célula de manufatura.

1.2.2 OBJETIVOS ESPECÍFICOS

A pesquisa possui os seguintes objetivos específicos:

- Investigar problemas relacionados ao escalonamento na produção (*job shop scheduling*);
- Investigar a TCS e suas extensões;
- Investigar ferramentas computacionais que auxiliem no cálculo de controle supervisório para Sistemas a Eventos Discretos Temporizados (SEDTs);
- Propor um método para realizar uma avaliação de desempenho de células de manufatura;
- Validar o método proposto em um exemplo de uma célula de manufatura, em uma indústria metalúrgica.

1.3 ORGANIZAÇÃO DO DOCUMENTO

Essa dissertação está estruturada conforme segue: no capítulo 2 é apresentada uma introdução à Teoria de Sistemas, algumas de suas sub-divisões, conceitos básicos dos Sistemas a Eventos Discretos e sua classificação dentro desse contexto.

O capítulo 3 apresenta a teoria de linguagens e autômatos, utilizada na modelagem de SEDs, a Teoria de Controle Supervisório, conforme proposta por Ramadge e Wonham (RAMADGE; WONHAM, 1989), e algumas de suas extensões: Controle Supervisório Modular (WONHAM; RAMADGE, 1988) e Modular Local (QUEIROZ; CURY, 2000), Controle Supervisório de Sistemas a Eventos Discretos Temporizados (BRANDIN; WONHAM, 1992), Controle Modular Temporizado (BRANDIN; WONHAM; BENHABIB, 1992).

O capítulo 4 apresenta conceitos da teoria de escalonamento na indústria de manufatura e algumas das abordagens para solucionar tais problemas.

O capítulo 5 apresenta a proposta de um método para a obtenção de escalonamento com o uso da TCS e SEDTs.

O capítulo 6 apresenta um problema de escalonamento em uma indústria de manufatura de elevadores, e a aplicação da proposta apresentada no capítulo 5.

No capítulo 7 são apresentadas as conclusões e sugestões para trabalhos futuros.

2 REVISÃO DA TEORIA DE SISTEMAS

O objetivo do capítulo é contextualizar o problema estudado dentro do conjunto dos sistemas, objetos de estudo da engenharia. Para tal, são apresentados alguns conceitos básicos da teoria, critérios de identificação e classificação, bem como a apresentação de distinções entre sistemas dinâmicos variados continuamente e sistemas dinâmicos discretos, tema principal deste trabalho. O desenvolvimento foi feito baseado em (CASSANDRAS; LAFORTUNE, 2008).

2.1 DEFINIÇÃO DE SISTEMAS

Um sistema pode ser definido, de uma maneira bastante genérica, como um conjunto de elementos, ou componentes, que interagem para desempenhar determinada função. Pode ainda, ser definido como uma combinação de componentes que trabalham em conjunto para realizar uma função que não seria possível com quaisquer uma das partes individuais (CASSANDRAS; LAFORTUNE, 2008). Essa definição é bastante abrangente, podendo ser aplicada a uma infinidade de contextos, e não precisa, necessariamente, descrever sistemas físicos, podendo, por exemplo, descrever a dinâmica de crescimento econômico de uma nação, ou seu grau de desenvolvimento (CASSANDRAS; LAFORTUNE, 2008).

2.1.1 MODELAGEM DE SISTEMAS

Os sistemas reais, físicos ou não, devido à quantidade de variáveis e, muitas vezes, com um comportamento que não pode ser descrito analiticamente, são representados por modelos. Os modelos auxiliam na análise de um sistema, e sua escolha, e o nível de precisão adotado, têm grande influência na maneira como um problema será abordado e analisado.

O processo de modelagem pode ser feito basicamente em quatro passos (CASSANDRAS; LAFORTUNE, 2008): i) especificar um conjunto de variáveis mensuráveis; ii) selecionar um subconjunto das variáveis, estabelecendo as variáveis de entrada; iii) selecionar outro subconjunto das variáveis, estabelecendo as variáveis de saída; iv) postular a existência de uma relação matemática entre as entradas e as saídas. Essas variáveis são então mensuradas ao longo de um determinado período de tempo $t_0 \leq t \leq t_f$, t_0 é o tempo inicial e t_f é o tempo final.

As variáveis de entrada podem ser representadas por um vetor coluna de funções do tempo, escrito na forma $u(t) = [u_1(t), \ldots, u_p(t)]^T$, e similarmente, as variáveis de saída podem ser representadas por $y(t) = [y_1(t), \ldots, y_m(t)]^T$, obtendo-se então o modelo do sistema na forma matemática $y(t) = g(u(t), t) = [g_1(u_1(t), \ldots, u_p(t)), \ldots, g_m(u_1(t), \ldots, u_p(t))]^T$, onde $g(u(t))$ é um vetor coluna, cujas entradas são as funções $g_1(.), \ldots, g_m(.)$.

Tais modelos podem ser utilizados para descrever e analisar processos em diversos sistemas, por exemplo, células de manufatura, nas quais a evolução das mesmas dependem de regras e especificações que definem as condições para que determinados eventos ocorram (CASSANDRAS; STRICKLAND, 1989).

Uma modelagem conforme descrita, está ilustrada na Figura 1.

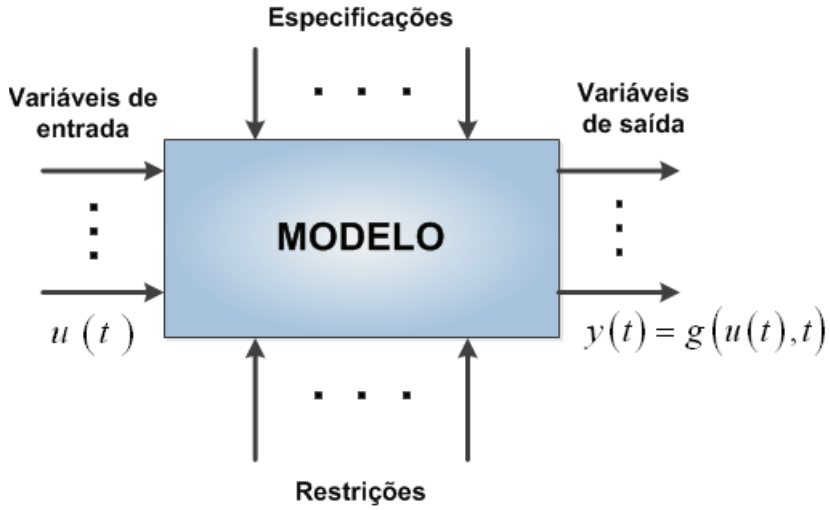

Figura 1 – Modelagem de um sistema.

2.1.2 SISTEMAS ESTÁTICOS E DINÂMICOS

As funções que relacionam as entradas e saídas de um sistema, podem ser simples funções algébricas, ou funções mais complexas, onde a saída é dependente do tempo, podendo ou não ser afetada pelo valor de $u(t)$ para $t = 0$. Um sistema é dito dinâmico se o valor $y(t)$ para algum $t > 0$ depende de valores passados de $u(t)$. Se a saída $y(t)$ independe de valores passados de $u(t)$, então o sistema é dito estático.

Os sistemas dinâmicos, objetos deste estudo, podem ser divididos em duas classes: i) sistemas variantes no tempo, e ii) sistemas invariantes no tempo ou sistemas estacionários (CASSANDRAS; LAFORTUNE, 2008).

Considerando-se um sistema invariante no tempo, e se, ao mesmo, for aplicado uma entrada $u_1(t)$, para a qual o sistema forneça uma saída $y_1(t)$, se à essa entrada for aplicado um atraso de tempo $u_2(t) = u_1(t-\tau)$, sua saída será então, uma versão com atraso de $y_1(t)$, ou seja $y_2(t) = y_1(t-\tau)$, com atraso igual ao de u_2 em relação a u_1. A figura 2 ilustra essa propriedade.

Para o sistema variante no tempo, se o mesmo atraso $u_2(t) = u_1(t-\tau)$ for aplicado, sua saída não será uma versão com atraso de $y_1(t)$, ou seja $y_2(t) \neq y_1(t-\tau)$.

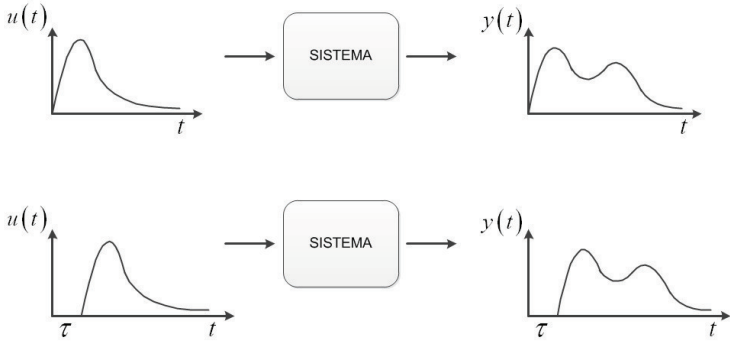

Figura 2 – Sistema invariante no tempo.

2.1.3 SISTEMAS LINEARES E NÃO LINEARES

Seja considerado um sistema com duas variáveis de entrada e duas variáveis de saída, cujo modelo é representado pela figura 3:

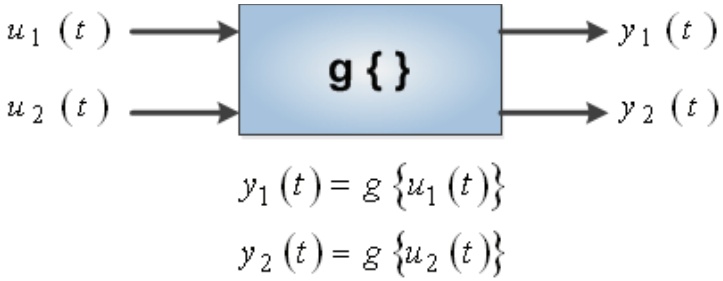

Figura 3 – Modelo de um sistema com duas entradas e duas saídas.

Os seguintes princípios podem ser definidos (YANG et al., 2009):

- Adição:

$$g\{u_1(t)+u_2(t)\} = g\{u_1(t)\}+g\{u_2(t)\} = y_1(t)+y_2(t) \tag{1}$$

- Homogeneidade ou escalonamento:

$$g\{u(t)\} = y(t) \Rightarrow g\{a\cdot u(t)\} = a\cdot g\{u(t)\} = a\cdot y(t), a \in \mathbb{R} \tag{2}$$

A associação do princípio da adição e da homogeneidade, resulta no princípio da sobreposição dos efeitos:

$$\begin{aligned} g\{a\cdot u_1(t)+b\cdot u_2(t)\} &= g\{a\cdot u_1(t)\}+g\{b\cdot u_2(t)\} \\ &= a\cdot g\{u_1(t)\}+b\cdot g\{u_2(t)\} \\ &= a\cdot y_1(t)+b\cdot y_2(t) \end{aligned} \tag{3}$$

Um sistema é dito linear, se e somente se atender ao princípio da sobreposição dos efeitos (CASSANDRAS; LAFORTUNE, 2008).

2.1.4 SISTEMAS ESTADOS CONTÍNUOS E SISTEMAS ESTADOS DISCRETOS

Outra subdivisão dos sistemas diz respeito à natureza do espaço de estados de um modelo. O estado de um sistema pode ser definido como um conjunto de variáveis, chamadas de variáveis de estado, tal que o conhecimento de tais variáveis em um dado instante de tempo $t = t_0$, juntamente com a entrada para $t \geq t_0$ determina completamente o comportamento do sistema para qualquer instante de tempo $t \geq t_0$ (OGATA, 2003).

No modelo de estados contínuos, o espaço de estados é composto por todos os vetores de números reais, com dimensão n. No modelo de estados discretos, o espaço de estados é um conjunto discreto, ou seja, as variáveis de estado passam de um estado a outro em instantes discretos no tempo. A observação do comportamento dinâmico de sistemas com essas características é simples, uma vez que a lógica de transição é baseada em regras do tipo “se em dado momento o estado é x, na ocorrência de algum evento o próximo estado passa então a ser x_1”.

A Figura 4 mostra um exemplo típico de um sistema estado discreto, representado por uma esteira de bagagens, comumente encontrada em aeroportos. As bagagens são depositadas na esteira, e ficam à disposição para retirada, pelos respectivos proprietários. A quantidade de bagagens à disposição na esteira $x(t)$ incrementa em uma unidade cada vez que um item é depositado, e quando uma bagagem é retirada, $x(t)$ decrementa em uma unidade.

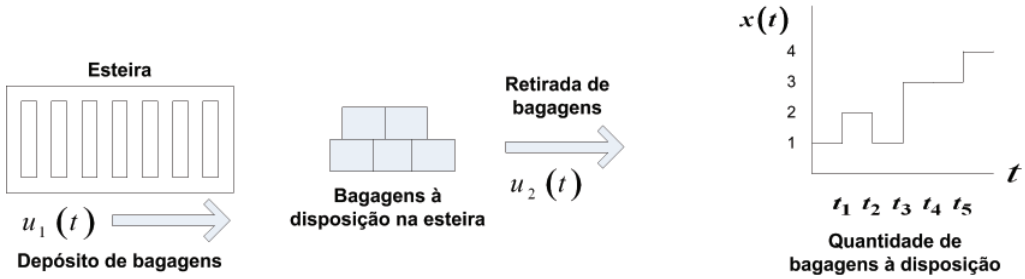

Figura 4 – Esteira de aeroporto - sistemas de estados discretos.

Um exemplo de um sistema estado contínuo poderia ser um tanque de água, com duas válvulas de entrada, uma para a água e outra para adição de compostos químicos utilizados no tratamento. Nesse caso, $x(t)$ poderia ser o valor da acidez (ph) da água tratada, analisada ao longo do tempo, e K representaria o valor nominal desejado. Esse exemplo é mostrado na Figura 5.

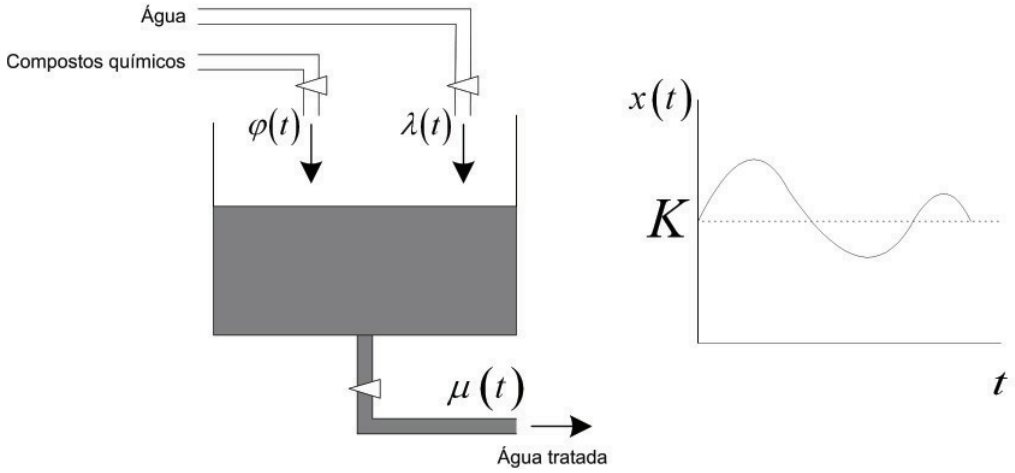

Figura 5 – Tanque de água - sistemas de estados contínuos.

2.1.5 SISTEMAS ESTOCÁSTICOS E DETERMINÍSTICOS

Considerando o tanque da Figura 5, onde a quantidade de água na entrada $\lambda(t)$, dependa por exemplo, da captação de precipitações pluviométricas. Poder-se-ia nesse caso, incorporar modelos probabilísticos ao modelo do sistema, de modo que se possa definir com certa precisão, quando e qual quantidade de água ter-se-ia no futuro. Ou então, no exemplo da Figura 4, a equação de estados do sistema, depende do conhecimento da função de entrada $u_1(t)$. Considerando que a entrada de bagagens na esteira, depende da chegada dos aviões de passageiros nos terminais, excesso de tráfego na pista, aeroportos interditadas ou condições climáticas podem influenciar no comportamento dessa variável. Novamente, uma modelagem com maior precisão seria possível somente com a utilização de modelos probabilísticos. Por intermédio de $u_1(t)$ e $u_2(t)$, o sistema seria descrito dinamicamente, porém, seus valores não seriam deterministicamente conhecidos para todo t. Outra classificação de sistemas pode então ser definida: um sistema é dito estocástico se, ao menos, uma de suas variáveis de estado variar aleatoriamente, caso contrário, é dito determinístico. A equação de estados de um sistema dinâmico é dito estocástica se sua evolução puder ser descrita somente com o uso de funções probabilísticas.

2.1.6 SISTEMAS A EVENTOS DISCRETOS

Em sistemas estado contínuo, os valores das variáveis de estado modificam continuamente de acordo com o tempo, que é uma variável independente, que aparece como argumento das funções de entrada e saída do sistema.

Os sistemas a eventos discretos diferem dos sistemas dinâmicos que variam continuamente, pois não podem ser modelados adequadamente por equações diferenciais ou equações a diferenças (CASSANDRAS; LAFORTUNE, 2008).

Se em determinado sistema as entradas e saídas puderem ser definidas somente em instantes discretos de tempo, ou seja, as transições de estados são observadas somente em pontos discretos no tempo, e essas transições são associadas com eventos, como resultado o chamado sistema tempo discreto é obtido. Um SED é, portanto, um sistema dinâmico que evolui de acordo com a ocorrência de eventos em intervalos de tempo, normalmente irregulares e desconhecidos (CURY, 2001), sendo definido como um sistema dirigido por eventos.

Em sistemas dirigidos pelo tempo, os estados mudam continuamente com o tempo. Neste caso, a variável tempo t é uma variável independente, que aparece como argumento de todas as funções de entrada, saída e de estados.

Inúmeros sistemas, notadamente os sistemas tecnológicos, são sistemas a eventos discretos, por exemplo:

- um sistema de manufatura, composto por um braço robótico que deve posicionar determinada peça em uma máquina de dobra. A evolução do processo de fabricação depende da ocorrência de eventos discretos ao longo do tempo: prender a peça, posicionar na máquina de dobra, executar processo de dobra, etc.;
- em um jogo de xadrez, a evolução da partida ocorre por intermédio de cada nova disposição das peças. Um novo estado do sistema é definido a cada jogada (evento).

2.2 SUMÁRIO DA CLASSIFICAÇÃO DE SISTEMAS

Neste capítulo foi apresentado um resumo da classificação de sistemas com o objetivo de se descrever as principais características dos sistemas a eventos discretos, objeto principal desse estudo:

- sistemas estáticos e dinâmicos. Em sistemas dinâmicos, as saídas dependem de valores passados da entrada;
- sistemas lineares e não lineares. Um sistema é dito linear, se e somente se satisfizer a propriedade de sobreposição dos efeitos;
- sistemas estados contínuos e estados discretos. Em sistemas estados discretos, as variáveis de estado são formadas por elementos de conjuntos de valores discretos enumeráveis;
- sistemas estocásticos e determinísticos. Se uma ou mais variáveis do sistema puderem assumir valores aleatórios, então o sistema é dito estocástico.
- sistemas a eventos discretos. A transição de estados ocorre na ocorrência de eventos discretos, gerados de modo assíncrono.

Segundo a classificação descrita, um SED é um sistema dinâmico, invariante no tempo, não linear, de estados discretos e dirigido por eventos.

A Figura 6 sumariza a classificação apresentada.

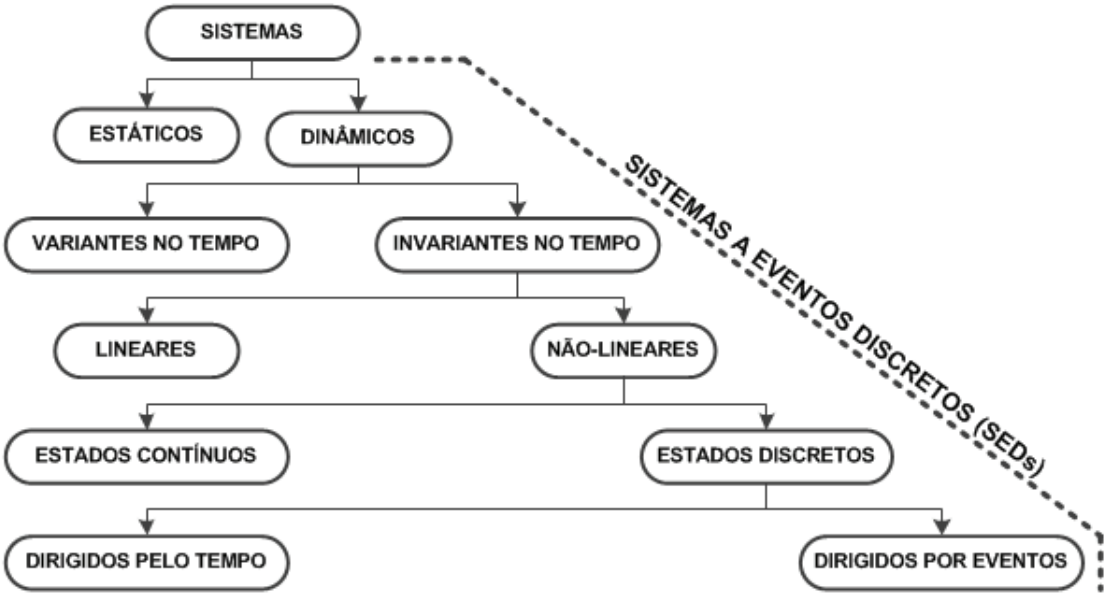

Figura 6 – Classificação de sistemas.

3 TEORIA DE CONTROLE SUPERVISÓRIO E EXTENSÕES

O constante desenvolvimento de tecnologia computacional tem levado os SEDs a níveis de complexidade tão elevados, que ferramentas formais mais detalhadas têm sido cada vez mais necessárias (RAMADGE; WONHAM, 1989), motivando os pesquisadores a trabalhar no desenvolvimento de modelos matemáticos apropriados, que descrevam o comportamento e que propiciem o desenvolvimento, controle e avaliação de desempenho de SEDs, tais como:

- Redes de Petri (MURATA, 1989);
- Teoria das Filas (KLEINROCK, 1975);
- Algebra Max-Plus (BACCELLI et al., 1992).

Várias das abordagens encontradas na literatura enfatizam a análise de sistemas já propostos, baseados principalmente no conhecimento e experiência dos projetistas (QUEIROZ; CURY, 2002a). A vantagem dos modelos das Redes de Petri controladas (HOLLOWAY; KROGH; GIUA, 1997) e do modelo de Ramadge e Wonham (RAMADGE; WONHAM, 1989), baseado em linguagens e autômatos, reside no fato de que estas contam com procedimentos formais de análise e síntese de controladores.

Com o modelo proposto por Ramadge e Wonham (1989), por intermédio da TCS, é possível a síntese automática de supervisores de modo que o mesmo atenda às especificações de controle e de restrições da planta (PAULA; SANTOS, 2008), (QUEIROZ; CURY, 2002a). Além disso, os autômatos também podem ser facilmente combinados de modo que sistemas mais complexos podem ser modelados a partir de componentes mais simples, de uma maneira bastante sistemática (CASSANDRAS; LAFORTUNE, 2008).

Por estas razões, o formalismo utilizado nesse trabalho, para o controle de SEDs é feito por intermédio do uso dos conceitos de linguagens e autômatos.

A modelagem de uma planta e de suas especificações de controle, tal que um supervisor resultante gere um controle minimamente restritivo fundamenta-se na teoria de controle supervisório desenvolvida por Ramadge e Wonham (RAMADGE; WONHAM, 1989).

O objetivo desse capítulo é apresentar a TCS e suas extensões, com o desenvolvimento feito baseado em (CURY, 2001), (WONHAM, 2011), (CASSANDRAS; LAFORTUNE, 2008), (BRANDIN; WONHAM, 1992), (BRANDIN; WONHAM; BENHABIB, 1992), (BRANDIN; WONHAM, 1993), (QUEIROZ; CURY, 2002a) e (SAADATPOOR, 2004).

Para realizar a modelagem das plantas, obtenção dos resultados das operações entre autômatos temporizados, bem como dos supervisores em SEDTs, neste estudo é utilizado o *software* TTCT (WONHAM, 2011), que foi desenvolvido na Universidade de Toronto, Canadá. Essa ferramenta estende as funcionalidades do sistema TCT (FENG; WONHAM, 2006), para sistemas a eventos discretos não temporizados, desenvolvida na mesma instituição.

3.1 CONCEITOS DE LINGUAGENS EM SEDS

Um SED pode ser descrito como sendo um sistema de estados discretos, dirigido por eventos, i.e., a evolução dos estados desse sistema, depende da ocorrência de eventos, em geral, assíncronos em relação à escala de tempo (CASSANDRAS; LAFORTUNE, 2008). Uma sequência de tais eventos pode ser escrita na forma $e_1 e_2 e_3 ... e_n$, e a qualquer SED pode-se então, associar um conjunto de eventos Σ, o qual pode ser entendido como o alfabeto de uma linguagem, e as sequências de eventos como as palavras (ou cadeias) de tal linguagem.

A linguagem é a maneira formal de representar os SEDs, e especifica todas as sequências possíveis de eventos que um SED pode processar, sem a necessidade de qualquer estrutura adicional.

3.1.1 DEFINIÇÕES BÁSICAS

Uma linguagem L, definida a partir de um conjunto de eventos Σ, é um conjunto de cadeias de tamanhos finitos formadas por justaposição a partir dos elementos (eventos) de Σ. Considerando o conjunto de eventos $\Sigma = \{a, b, g\}$, as seguintes linguagens podem ser definidas:

- $L_1 = \{\varepsilon, a, abb\}$; ou
- $L_2 = \{$todas as possíveis cadeias de comprimento 3$\}$; ou
- $L_3 = \{$todas as possíveis cadeias de comprimento finito que comecem com o evento $a\}$.

onde ε é a cadeia vazia que não é composta por nenhum evento.

O conjunto de todas as cadeias possíveis formadas por elementos do conjunto Σ, inclusive a cadeia vazia ε, é chamado de fechamento *Kleene*, e é representado por Σ^*, e qualquer linguagem sobre Σ é um subconjunto de Σ^*.

Seja a cadeia $s = tuv$, com $s, t, u, v \in \Sigma$, os seguintes termos podem ser definidos:

- t é chamado de prefixo de s;
- u uma subcadeia de s;
- v é chamado de sufixo de s.

3.1.2 PROPRIEDADES DE LINGUAGENS

Sendo uma linguagem um conjunto, portanto todas as operações de conjunto como união, interseção, diferença e complemento são aplicáveis. Adicionalmente, as seguintes propriedades podem ser definidas:

Concatenação: é uma das principais operações na formação de cadeias, e consequentemente, de linguagens. A cadeia abb por exemplo, é a concatenação da cadeia ab com o evento b.

Se $L_1, L_2 \subseteq \Sigma^*$ logo:

$$L_1 L_2 = \{s_1 s_2 \in \Sigma^* : (s_1 \in L_1)\,(s_2 \in L_2)\} \tag{4}$$

ou seja, uma cadeia está em $L_1 L_2$ se puder ser escrita como uma concatenação de uma cadeia em L_1 e uma cadeia em L_2.

Prefixo fechamento: seja $L \subseteq \Sigma^*$, o prefixo fechamento é definido por:

$$\overline{L} = \{s \in \Sigma^* : (\exists t \in \Sigma^*)\,[st \in L]\} \tag{5}$$

$\overline{L}$ consiste em todos os prefixos de todas as cadeias em L. Em geral $L \subseteq \overline{L}$. Uma linguagem L é dita prefixo-fechada se qualquer prefixo de qualquer cadeia em L é também um elemento de $\overline{L}$.

Fechamento *Kleene*: é a mesma operação definida para o conjunto Σ, porém aplicado a L, cujos elementos podem ser cadeias com comprimento maior que 1. Um elemento de L^* é formado pela concatenação de um número finito de elementos de L. Isto inclui a concatenação de elementos vazios, ou seja, a cadeia ε:

$$L^* = \{\varepsilon\} \cup L \cup LL \cup LLL \cup \ldots \tag{6}$$

Considerando $\Sigma = \{a, b, g\}$ e duas linguagens definidas sobre Σ: $L_1 = \{\varepsilon, a, bgg\}$ e $L_4 = \{g\}$. L_1 e L_4 não são prefixo-fechadas pois $b, bg \notin L_1$ e $\varepsilon \notin L_4$, e:

$L_1 L_4 = \{g, ag, bggg\}$
$\overline{L_1} = \{\varepsilon, a, b, bg, bgg\}$
$\overline{L_4} = \{\varepsilon, g\}$
$L_1 \overline{L_4} = \{\varepsilon, a, bgg, g, ag, bggg\}$
$L_4^* = \{\varepsilon, g, gg, ggg, \ldots\}$
$L_1^* = \{\varepsilon, a, bgg, aa, abgg, bgga, bggbgg, \ldots\}$

3.1.3 LINGUAGEM GERADA E LINGUAGEM MARCADA

Em um determinado sistema S, onde Σ representa o conjunto de todos os eventos possíveis em S, $L \subseteq \Sigma^*$ representa o conjunto de todas as cadeias de eventos possíveis de ocorrerem, linguagem gerada. $L_m \subseteq L$ é a chamada linguagem marcada, que representa o conjunto das cadeias em L que correspondem a tarefas completas executadas pelo sistema. Dessa maneira, o comportamento do sistema S pode ser completamente descrito por um par (L, L_m), e as seguintes propriedades são válidas:

- $L \supset L_m$, a linguagem marcada está contida na linguagem gerada pelo sistema;
- $L = \overline{L}$, a linguagem gerada por S é prefixo-fechada.

3.2 AUTÔMATOS

Um autômato é uma representação computável de uma linguagem, de acordo com regras bem definidas. Pode-se tomar como exemplo, o diagrama de transição de estados da Figura 7, onde os conjuntos de estados de um sistema são representados pelos nós $X = \{1,2,3\}$. Os arcos representam as funções de transições entre esses estados, ou seja, $f : X \times \Sigma \to X$:

$$f(1,\gamma) = 3, f(1,\lambda) = 3, f(2,\beta) = 3, f(2,\gamma) = 2, f(2,\alpha) = 1 \tag{7}$$

A notação $f(1,\gamma) = 3$ significa que, se o autômato estiver no estado 1, na ocorrência de um evento γ, uma transição instantânea para o estado 3 ocorre.

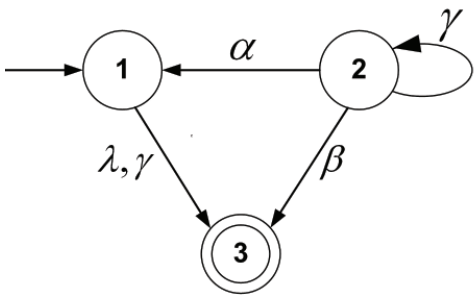

Figura 7 – Diagrama de transição de estados.

3.2.1 DEFINIÇÕES

No diagrama da Figura 7 pode-se observar o conjunto de eventos $\Sigma = \{\alpha, \beta, \gamma\}$, e o conjunto de estados $X = \{1, 2, 3\}$. O estado inicial do sistema 1 é marcado por uma seta, o estado marcado 3, indicando tarefa completa, é representado por um círculo duplo, ainda:

- um evento pode ocorrer sem mudança de estado $f(2, \gamma) = 2$;
- dois eventos distintos podem ocorrer a um estado causando exatamente a mesma transição $f(1, \lambda) = f(1, \gamma) = 3$;
- a função f é parcial no domínio $X \times \Sigma$, ou seja não há a necessidade de uma transição ser definida para cada evento em Σ em um estado em X.

3.2.2 AUTÔMATO DETERMINÍSTICO DE ESTADO FINITO

Um Autômato Determinístico de Estado Finito (ADEF) , representado por G , é uma quíntupla:

$$G = (X, \Sigma, f, x_0, X_m) \tag{8}$$

onde:
X é o conjunto de estados do autômato;
Σ é o conjunto de eventos do alfabeto;
$f : X \times \Sigma \to X$ é a função de transição;
x_0 é o estado inicial do autômato;
X_m é o conjunto de estados finais ou estados marcados, $X_m \subseteq X$.

É dito determinístico, pois f é uma função $X \times \Sigma \to X$ e não podem existir duas transições diferentes que ocorram a partir de um único estado, na ocorrência de um único evento levando a estados diferentes.

A operação do autômato G inicia no estado x_0 e, na ocorrência de um evento $e \subseteq \Sigma$, uma transição para o estado $f(x_0, e) \in X$ é executada. Esse processo continua baseada nas transições para as quais f é definida.

Para o processamento de uma cadeia de eventos, é utilizada uma extensão da função de transição f, representada por $\widehat{f}$, estendedo-se o domínio $f : X \times \Sigma \to X$ para $\widehat{f} : X \times \Sigma^* \to X$ (CASSANDRAS; LAFORTUNE, 2008), tal que:

$$\widehat{f}(x, \varepsilon) = x, \forall x \in \Sigma \tag{9}$$

$$\widehat{f}(x, se) := \widehat{f}(f(x, s), e) \text{ para } e \in \Sigma \text{ e } s \in \Sigma^* \tag{10}$$

3.2.3 PROPRIEDADES DOS AUTÔMATOS

Acessibilidade e co-acessibilidade: seja um ADEF definido por (8). Um estado $x \in X$ é dito ser acessível se $x = \widehat{f}(x_0, u)$ para algum $u \in \Sigma^*$, ou seja, partindo de um estado x_0, existe uma cadeia de eventos que leve o sistema de x_0 ao estado x, e se, para todo $x \in X$ tal condição for satisfeita, o autômato é dito acessível. Ainda, se para todo $x \in X$ existir uma cadeia de eventos $u \in \Sigma^*$ que leve o autômato para um estado marcado, o autômato é dito co-acessível.

A condição de co-acessibilidade pode ser representada por (CURY, 2001):

$$L(G) = \overline{L_m(G)} \tag{11}$$

Se as duas condições, acessibilidade e co-acessibilidade, forem satisfeitas, o autômato é dito $trim$, ou não-bloqueante. Por exemplo, o autômato G da figura 8 é dito bloqueante, pois uma vez no estado 4, nenhum estado marcado pode ser acessado, e G é dito não co-acessível. O estado 4 de G é dito não-acessível.

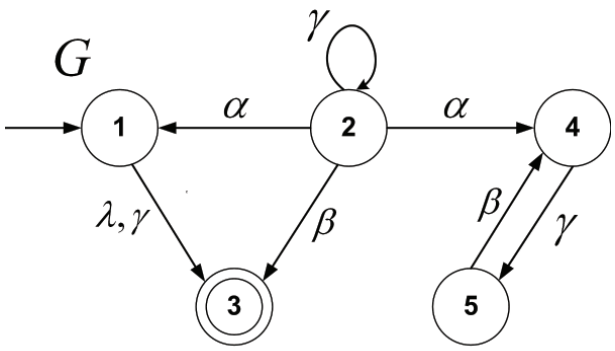

Figura 8 – Acessibilidade e co-acessibilidade.

Composição síncrona: sejam dois autômatos $G_1 = (X, \Sigma_1, f_1, x_0, X_m)$ e $G_2 = (Q, \Sigma_2, f_2, q_0, Q_m)$. A composição síncrona de G_1 e G_2 é representada por $G_1 \| G_2$, tal que:

$$G_1 \| G_2 = A_c \left(\Sigma_1 \cup \Sigma_2, X \times Q, f_{1\|2}, (x_0, q_0), X_m \times Q_m \right) \tag{12}$$

onde:

$$f_{1\|2}((x,q),\sigma) = \begin{cases} (f_1(x,\sigma), f_2(q,\sigma)) & \text{se } \sigma \in \Sigma_1 \cap \Sigma_2 \text{ e } f_1(x,\sigma), f_2(q,\sigma) \text{ definidas} \\ (f_1(x,\sigma), q) & \text{se } \sigma \in \Sigma_1 \text{ e } \sigma \notin \Sigma_2 \text{ e } f_1(x,\sigma) \text{ definida} \\ (x, f_2(q,\sigma)) & \text{se } \sigma \in \Sigma_2 \text{ e } \sigma \notin \Sigma_1 \text{ e } f_2(q,\sigma) \text{ definida} \\ \text{indefinida} & \text{caso contrário} \end{cases}$$

Caso exista um evento comum aos dois autômatos, o mesmo só é executado se ambos os autômatos o executarem simultaneamente, i.e., as transições dos dois autômatos deve estar sempre sincronizada em um evento comum, ou seja, um evento em $\Sigma_1 \cap \Sigma_2$.

Os estados de $G_1 \| G_2$ são normalmente representados aos pares, onde o primeiro componente é o estado atual de G_1 e o segundo componente, o estado atual de G_2. A linguagem do produto $G_1 \| G_2$, é caracterizada por:

$$L(G_1 \| G_2) = L(G_1) \| L(G_2) \tag{13}$$

$$L_m(G_1 \| G_2) = L_m(G_1) \| L_m(G_2) \tag{14}$$

Se $\Sigma_1 \cap \Sigma_2 = \emptyset$, então $L(G_1 \| G_2) = \{\varepsilon\}$, e $L_m(G_1 \| G_2)$ poderá ser $\emptyset$ ou $\{\varepsilon\}$, dependendo se estado inicial (x_{01}, x_{02}) for marcado ou não (CASSANDRAS; LAFORTUNE, 2008).

A figura 9 mostra um exemplo da composição síncrona.

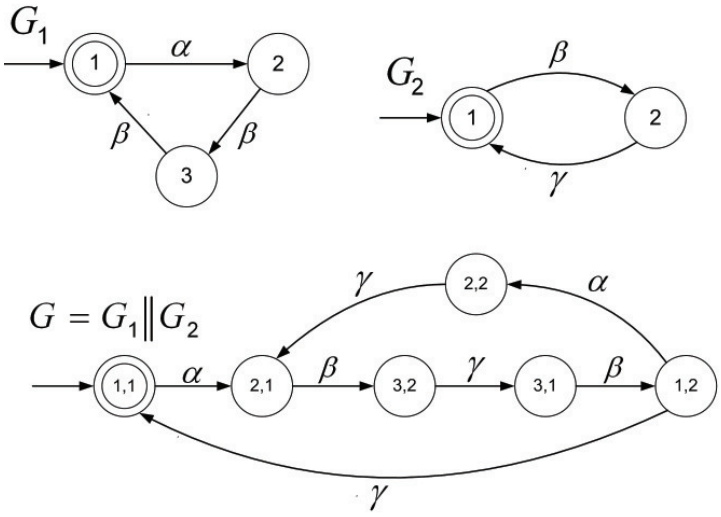

Figura 9 – Composição síncrona de dois autômatos.

O autômato G da figura 9 é o resultado do produto entre os autômatos G_1 e G_2. O conjunto de eventos comuns é $\{\beta\}$. Os estados do autômato G são compostos por um estado de G_1 e um estado de G_2, necessariamente nesta ordem. A partir do estado inicial $(1,1)$, G_1 pode executar o evento α, sem a participação de G_2, levando $G_1 \| G_2$ para o estado $(2,1)$. Após essa ocorrência, G_1 está no estado 2, e G_2 permanece no estado 1.

No estado $(2,1)$, o evento comum β é o único possível de ocorrer, levando o autômato de $(2,1)$ para $(3,2)$. O processo é repetido, encontrando-se todas as possíveis transições em todos os estados novos gerados.

O comando do *software* TTCT utilizado para obter a composição síncrona, é a função *Sync*, que faz a sincronização dos eventos em comum:

$$G = Sync\,(G_1, G_2) \tag{15}$$

Composição *Meet*:

O comando $Meet$ é um caso especial de $Sync$, onde o alfabeto considerado é dado por $\Sigma = \Sigma_1 \cup \Sigma_2$, e todos os eventos são compartilhados e a sincronização é total. Caso algum evento não ocorra em G_1 e G_2, o mesmo é bloqueado por $Meet$.

Considerando dois alfabetos Σ_1 e Σ_2, e dois SEDs G_1 e G_2, o procedimento $Meet$ retorna $G = Meet(G_1, G_2)$, acessível, de tal forma que:

$$L(G) = L(G_1) \cap L(G_2) \tag{16}$$

$$L_m(G) = L_m(G_1) \cap L_m(G_2) \tag{17}$$

A figura 10 mostra um exemplo do comando *Meet*:

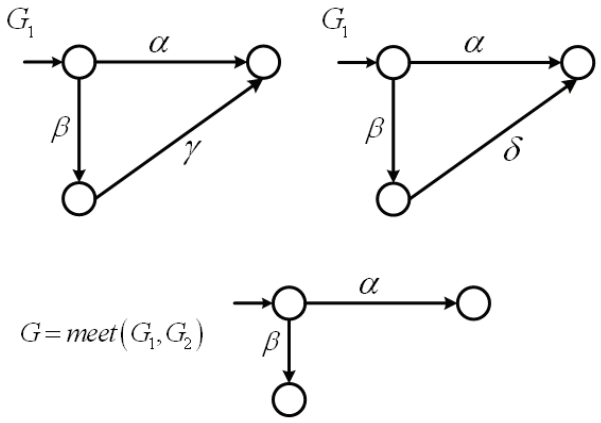

Figura 10 – Exemplo do comando $Meet$.

3.3 LINGUAGENS REPRESENTADAS POR AUTÔMATOS

Observando o diagrama de transição de estados de um autômato da figura 7, e considerando todos os caminhos que podem ser seguidos, partindo do estado inicial e que terminam em um estado marcado, as duas linguagens que podem ser associadas a um autômato são obtidas: a linguagem gerada $L(G)$ e a linguagem marcada $L_m(G)$.

A linguagem gerada por $G = (X, \Sigma, f, x_0, X_m)$ é:

$$L(G) := \left\{ s \in \Sigma^* : \widehat{f}(x_0, s) \text{ é definida} \right\} \tag{18}$$

A linguagem marcada é:

$$L_m(G) := \left\{ s \in L(G) : \widehat{f}(x_0, s) \in X_m \right\} \tag{19}$$

Um SED pode ser, dessa maneira, modelado por: i) um autômato G; ii) uma linguagem $L(G)$, que representa todas as cadeias de eventos que podem ocorrer em um autômato, partindo de um estado inicial, e iii) uma linguagem $L_m(G)$ que considera todas as cadeias que partindo de um estado inicial chegam a um estado marcado, ou seja, representa o conjunto de tarefas completas do sistema.

3.4 SUPERVISORES

Na abordagem desenvolvida por Ramadge e Wonham, a TCS apresenta um método de síntese de um supervisor ótimo minimamente restritivo e não bloqueante, para controlar um sistema, onde o SED é modelado por linguagens formais e autômatos (RAMADGE; WONHAM, 1989), (CURY, 2001).

O SED a ser controlado refere-se a um conjunto de unidades ou células comumente chamado de planta, representada por G. O comportamento de G pode ser caracterizado por $L(G)$, linguagem gerada, que são todas as sequências de eventos que podem ser geradas por G, e $L_m(G)$, linguagem marcada, que são todas as sequências de eventos que representam tarefas completas. Ao introduzir-se um controle, chamado de supervisor, é possível limitar o comportamento de G, de modo que se tenha um máximo desempenho, sem qualquer bloqueio e com restrições mínimas de $L(G)$.

A função do supervisor, denotado por S, é exercer uma ação de controle sobre a planta, acompanhando a execução dos eventos e determinando quais sequências devem ser evitadas, a fim de garantir a segurança e o comportamento do sistema segundo um conjunto de especificações. Essas especificações limitam o funcionamento, evitando que estados indesejados sejam alcançados, por exemplo, uma colisão entre um robô e um dispositivo pneumático de posicionamento de peças.

A supervisão da planta ocorre segundo uma estrutura em malha fechada, mostrada na Figura 11, onde o supervisor define uma entrada de controle habilitando determinado conjunto de eventos que podem ocorrer, e desabilitando os eventos indesejáveis para o sistema.

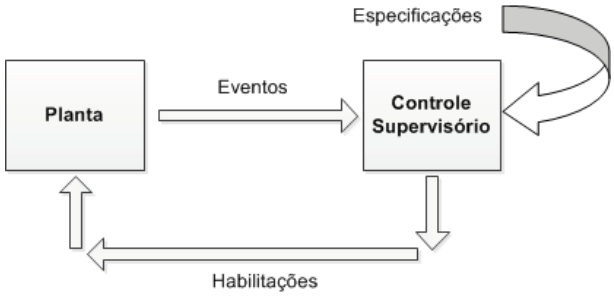

Figura 11 – Controle supervisório em malha fechada.

3.4.1 SUPERVISOR MONOLÍTICO

Para associar as estruturas de controle a uma planta G, o alfabeto de eventos Σ é dividido em (CASSANDRAS; LAFORTUNE, 2008):

- um subconjunto Σ_c de eventos controláveis, cuja ocorrência pode ser inibida ou desabilitada e;
- um subconjunto Σ_{unc} de eventos não controláveis, os quais estão sempre habilitados e podem sempre ocorrer sem qualquer intervenção do supervisor S.

Sobre esses eventos, define-se uma estrutura de controle Γ para G, e uma entrada de controle $\gamma \in \Gamma$, tal que:

$$\Gamma = \left\{ \gamma \in 2^{\Sigma} : \gamma \supseteq \Sigma_u \right\} \tag{20}$$

onde $\gamma \supseteq \Sigma_{unc}$ indica que os eventos não controláveis não podem ser desabilitados.

O supervisor S é um autômato, definido sobre o alfabeto Σ, em que as mudanças de estado ocorrem de acordo com a sequência de eventos na planta G. Para cada estado da planta, o supervisor S executa uma ação de controle, desabilitando em G os eventos que não podem ocorrer em S após uma sequência de eventos observada. O supervisor S tem então, as funções de restringir o comportamento da planta e também de desmarcar estados, e uma tarefa será considerada completa se for marcada tanto pela planta quanto pelo supervisor, chamado então de supervisor marcador - *marking nonblocking supervisory controller* (MNSC) (WONHAM, 2011).

A planta G controlada por S é denotada por S/G, cujo comportamento é descrito pela composição síncrona de S e G $(S\|G)$.

O comportamento em malha fechada do SED, é caracterizado pelas seguintes linguagens:

$$L(S/G) = L(S\|G) = L(S)\|L(G) \tag{21}$$

$$L_m(S/G) = L_m(S\|G) = L_m(S)\|L_m(G) \tag{22}$$

e o supervisor S é dito não-bloqueante se $L(S/G) = \overline{L_m(S/G)}$ ($\overline{L}$ é o conjunto de todas as cadeias de Σ^* que são prefixos de cadeias de L).

Considerando-se uma planta G, com comportamento descrito por $L(G)$ e $L_m(G)$, uma estrutura de controle Γ, e uma linguagem $K \subseteq L(S/G)$, K é dita controlável em relação a G se, e somente se, $\overline{K}\Sigma_{unc} \cap L(G) \subseteq \overline{K}$. Ou seja, mesmo com a ocorrência de um evento não controlável no sistema, após uma cadeia de K, não gera uma sequência que desrespeite a especificação.

Caso a linguagem K não seja controlável, é possível obter uma aproximação de K, chamada de máxima linguagem controlável contida em K, e que é denotada por $supC(K)$, e o supervisor que implementa esta linguagem é chamado supervisor não bloqueante e minimamente restritivo, tal que $L_m(S/G) = supC(K)$ (RAMADGE; WONHAM, 1989).

O supervisor obtido é tido como sendo monolítico, pois há somente um único supervisor atuando sobre a planta, de modo que todas as especificações sejam atendidas, porém o número de estados da planta G ou das especificações crescem exponencialmente com o número de subsistemas ou de restrições de coordenação, podendo causar problemas de processamento em sistemas de grande porte.

Tal complexidade pode ser reduzida com a abordagem do controle supervisório modular (QUEIROZ; CURY, 2002a), em que o supervisor monolítico é substituído por supervisores modulares, sem perda de desempenho. Além de o cálculo ser mais simples, a manutenção, modificação e atualização tornam-se menos complexas. Por exemplo, caso seja necessária qualquer alteração em alguma tarefa, apenas o supervisor afetado pela nova especificação precisa ser novamente avaliado, tornando a abordagem modular mais flexível, quando comparada com a abordagem monolítica (BRANDIN; WONHAM, 1993).

Porém, no controle supervisório modular local (QUEIROZ; CURY, 2000), cada supervisor é aplicado sobre uma planta local, e cada planta pode ter interferência com outras, significando que elas compartilham eventos. O compartilhamento de eventos pode causar bloqueio quando mais de um supervisor é aplicado simultaneamente sobre a planta, neste caso, os supervisores são considerados conflitantes (PENA et al., 2010), tornando-se necessária a aplicação de testes para a verificação da condição de não-conflito entre os diversos supervisores.

3.4.2 SUPERVISOR MODULAR

O crescimento exponencial em número de estados, observado no supervisor monolítico, pode ser contornado por intermédio da abordagem proposta por Queiroz e Cury (QUEIROZ; CURY, 2002a) e (QUEIROZ; CURY, 2000), o Controle Modular Local (CML) e a Representação por Sistema Produto (RSP).

A planta pode ser subdividida em diversos subsistemas, com suas respectivas especificações de controle. Um supervisor modular pode ser sintetizado para cada especificação, de modo que a atuação em conjunto dos diversos supervisores satisfaça a especificação global do sistema. Ou seja, a ação de controle é dividida entre os diversos supervisores locais, cada um representando a máxima linguagem controlável para cada subsistema.

A Figura 12 ilustra o funcionamento de um supervisor modular:

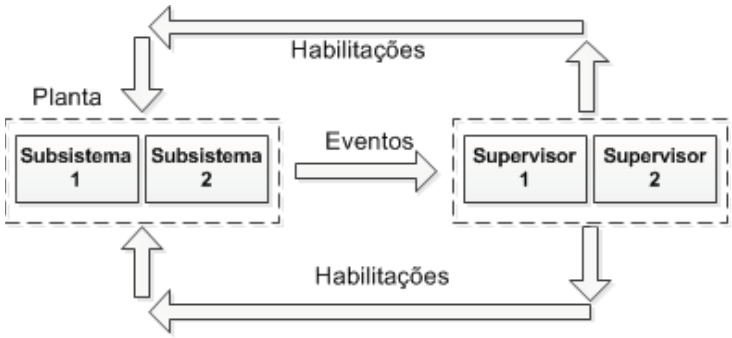

Figura 12 – Controle modular local.

Cada um dos diversos subsistemas de uma planta pode ser representado por um ADEF:

$$G_i = (\Sigma_i, Q_i, \delta_i, q_{0i}, Q_{mi}), i \in I = \{1...n\} \tag{23}$$

onde n é o número subsistemas físicos, $\Sigma_i = \Sigma_{c_i} \cup \Sigma_{unc_i}$, e o alfabeto do sistema global Σ é obtido pela união dos diversos subconjuntos de alfabetos.

A RSP consiste na divisão de uma planta, no maior número possível de subsistemas completamente assíncronos entre si, ou seja, que não tenham eventos em comum, cada qual com um número mínimo de estados (QUEIROZ; CURY, 2002b).

Cada especificação genérica é representada por um autômato $E_{gen,j}$, com $j \in J = 1,...,m$, sendo m o número total das especificações, definidas em um subconjunto de eventos $\Sigma_{gen,j} \subseteq \Sigma$.

A composição síncrona dos diversos subsistemas, cujos comportamentos são restringidos por uma determinada especificação, resulta nas chamadas *plantas locais*, denotadas por $G_{loc,j}$. A planta local $G_{loc,j}$ associada à especificação $E_{gen,j}$ é definida então por:

$$G_{loc,j} = \|_{i \in I_{loc,j}} G_i \tag{24}$$

em que $I_{loc,j} = \left\{k \in I \,\middle|\, \Sigma_k \cap \Sigma_{gen,j} \neq \emptyset\right\}$, e a planta local $G_{loc,j}$ é composta pelos subsistemas que estão restringidos por $E_{gen,j}$, e a planta reduzida ou enxuta $G_e = \left\|_{j=1}^{m} G_{loc,j}\right.$ é composta apenas dos subsistemas relevantes ao problema de controle.

As especificações locais podem ser expressas em termos das plantas locais $G_{loc,j}$:

$$E_{loc,j} = E_{gen,j} \left\| L_m \left(G_{loc,j} \right) \right. \tag{25}$$

As máximas linguagens controláveis locais são obtidas com:

$$S_j = SupC(E_{loc,j}, G_{loc,j}) \tag{26}$$

Após a obtenção das máximas linguagens controláveis locais, é possível a realização de um supervisor local para cada uma das especificações, e se o conjunto $SupC(E_{loc,j}, G_{loc,j})$ for localmente modular, isto é, se forem não conflitantes:

$$\left\| _j \overline{SupC\left(E_j, G_j\right)} = \overline{\left\| _j SupC\left(E_j, G_j\right)\right.} \right. \tag{27}$$

ou seja, se a modularidade local dos supervisores locais for verificada, a ação conjunta de todos os supervisores locais é não bloqueante, e o supervisor monolítico pode ser substituído por supervisores modulares, sem perda de desempenho (QUEIROZ; CURY, 2002a):

$$SupC\left(\cap_{i=1}^{n} E_{global,i}, G_e\right) = \left\|_{i=1}^{n} SupC\left(E_{loc,i}, G_{loc,i}\right)\right. \tag{28}$$

3.5 SISTEMAS A EVENTOS DISCRETOS TEMPORIZADOS

A representação de um SED pode ser dividida em dois modelos: temporizados e não temporizados. No modelo não temporizado, somente o comportamento lógico e a evolução da sequência de estados de um sistema são considerados. A duração de cada evento ou da transição e do tempo que o sistema permanece em determinado estado não são avaliados. No modelo temporizado, além do comportamento lógico e sequencial, as informações de tempo são também avaliadas.

Entre as alternativas para a modelagem de SEDTs, se destacam os modelos temporizados de Brandim e Wonham (BRANDIN; WONHAM; BENHABIB, 1992), utilizados no desenvolvimento deste trabalho.

3.5.1 REPRESENTAÇÃO DE SEDTS

Seja uma quíntupla na forma:

$$G_{act} = (A, \Sigma_{act}, \delta_{act}, a_0, A_m) \tag{29}$$

onde:
A é o conjunto de atividades do autômato;
Σ_{act} é o conjunto finito de eventos;
$\delta_{act} : A \times \Sigma_{act} \rightarrow A$ é a função de transição;
a_0 é a atividade inicial do autômato;
$A_m \subseteq A$ é o conjunto de atividades finais ou atividades marcadas.

Do mesmo modo que a função transição de eventos f em 8, a função transição de atividades δ_{act} é também uma função parcial $\delta_{act} : \Sigma_{act} \times A \rightarrow A$. Uma transição de atividades é uma tripla $[a, \sigma, a']$ com $a' = \delta_{act}(\sigma, a)$.

Para a construção de um modelo para um SEDT, informações de tempos das atividades são adicionadas em G_{act}. Considerando $\mathbb{N}$ o conjunto de todos os numerais inteiros e não negativos, no alfabeto de eventos δ_{act} a cada transição σ estão associados um limite de tempo inferior $l_\sigma \in \mathbb{N}$ e um limite de tempo superior $u_\sigma \in \mathbb{N} \cup \{\infty\}$, tal que $l_\sigma \leq u_\sigma$. Esses limites de tempo correspondem aos tempos mínimos e máximos para a ocorrência do evento σ se o mesmo estiver habilitado. Por convenção, se o limite de tempo superior é igual a ∞ significa que o evento pode ocorrer a qualquer instante.

O conjunto de eventos temporizados Σ_{act} pode ser particionado em 2 subconjuntos:

$$\Sigma_{act} = \Sigma_{spe} \cup \Sigma_{rem} \tag{30}$$

onde $spe =$prospectivo e $rem =$ remoto.

Se um evento σ é prospectivo, u_σ é finito e $0 \leq u_\sigma < \infty$ e $0 \leq l_\sigma \leq u_\sigma$, se σ é remoto, u_σ=∞ , e $0 \leq l_\sigma \leq \infty$. O conjunto $(\sigma, l_\sigma, u_\sigma)$ é chamado de “evento temporizado”:

$$\Sigma_{tim} := \{(\sigma, l_\sigma, u_\sigma) | \, \sigma \in \Sigma_{act}\} \tag{31}$$

Para cada $\sigma \in \Sigma_{act}$, o intervalo do temporizador T_σ é definido por:

$$T_\sigma \begin{cases} [0, u_\sigma] \text{ se } \sigma \in \Sigma_{spe} \\ [0, l_\sigma] \text{ se } \sigma \in \Sigma_{rem} \end{cases} \tag{32}$$

Um SEDT, conforme Brandin e Wonham (BRANDIN; WONHAM, 1992), pode ser definido por uma quíntupla:

$$G = (Q, \Sigma, \delta, q_0, Q_m) \tag{33}$$

- Q=$A \times \Pi\{T_\sigma | \sigma \in \Sigma_{act}\}$;
- $q \in Q$, tal que $q = \{a, \{t_\sigma | \sigma \in \Sigma_{act}\}\}$, e $a \in A, t_\sigma \in T_\sigma$;
- q_0 é o estado inicial, dado por $q_0 = \{a_0, \{t_{\sigma 0} | \sigma \in \Sigma_{act}\}\}$, onde:

$$t_{\sigma 0} \begin{cases} u_\sigma, \text{ se } \sigma \in \Sigma_{spe} \\ l_\sigma, \text{ se } \sigma \in \Sigma_{rem} \end{cases} \tag{34}$$

- Q_m é o conjunto de estados marcados, dado por $Q_m \subseteq A_m \times \Pi\{T_\sigma | \sigma \in \Sigma_{act}\}$;
- $tick$ é o evento que representa a ocorrência de uma unidade de tempo do relógio global;
- o conjunto de eventos $\Sigma = \Sigma_{act} \cup (tick)$;
- função de transição δ:$\Sigma \times Q \to Q$.

Para qualquer $\sigma \in \Sigma$ e para qualquer $q = (a, \{t_\tau | \tau \in \Sigma_{act}\}) \in Q$, a função de transição $\delta(q, \sigma)$ é definida (representado por $\delta(q, \sigma)!$), se e somente se (SAADATPOOR, 2004):

- $\sigma = tick$ e $\forall \tau \in \Sigma_{spe}; \delta_{act}(a, \tau)! \Rightarrow t_\tau > 0$,

  i.e., nenhum prazo limite de um evento prospectivo é igual a 0;

- $\sigma \in \Sigma_{spe}$, e $\delta_{act}(a, \tau)!$ e $0 \leq t_\sigma \leq u_\sigma - l_\sigma$,

  i.e., se σ é definido para uma atividade a em G_{act} e o atraso para sua ocorrência já tenha sido atingido, σ pode ocorrer, e deve ocorrer antes do seu prazo limite;

- $\sigma \in \Sigma_{rem}$, e $\delta_{act}(a,\tau)!$ e $t_\sigma = 0$,

  i.e., se σ é definido para uma atividade a em G_{act} e o atraso para sua ocorrência já tenha sido atingido, σ pode ocorrer.

Para completar a definição geral de um SEDT, uma condição deve ser imposta, de modo a se excluir uma situação, não realística fisicamente, em que o evento do relógio global *tick* é preterido por tempo indeterminado. Isto é, uma situação em que um ciclo de atividades ocorre infinitamente em um intervalo finito de tempo (BRANDIN; WONHAM, 1992). Tal condição é chamada de atividade cíclica.

Um SEDT possui uma condição de atividade cíclica se:

$$(\exists q \in Q)\,(\exists s \in \Sigma_{act})\,\delta(q,s) = q \tag{35}$$

ou seja, uma atividade ocorre indefinidamente, dentro de um intervalo de tempo fixo.

Essa situação deve ser evitada, tal que:

$$(\exists q \in Q)\,(\exists s \in \Sigma_{act})\,\delta(q,s) \neq q \tag{36}$$

A figura 13 mostra um exemplo de uma atividade cíclica, em uma planta G, onde no estado 1, a sequência de atividades $\{\alpha,\beta\}$ pode ocorrer um número infinito de vezes, sem a ocorrência de qualquer evento *tick*.

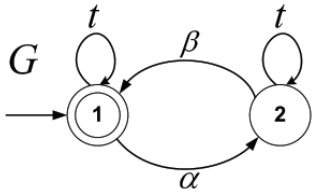

Figura 13 – Exemplo de uma atividade cíclica.

Definida a estrutura de transições de um SEDT, o comportamento de G em 33 pode ser caracterizado: seja Σ^* o conjunto de todos os elementos em Σ, incluindo a cadeia vazia ε. A função δ pode ser estendida para $\widehat{\delta} : Q \times \Sigma^* \to Q$, e a linguagem gerada $L(G)$ de G é o subconjunto de todas as cadeias em Σ^*, que podem ser geradas a partir da iteração de $\widehat{\delta}$, a partir de q_0 (todas as cadeias s, tal que $\widehat{\delta}(q_0,s)!$). E a linguagem marcada $L_m(G)$ de G, é o subconjunto de todas as cadeias em $L(G)$, para os quais o estado final pertence a Q_m (todas as cadeias s, tais que $\widehat{\delta}(q_0,s) \in Q_m$).

O comportamento de G é então definido por:

$$L(G) := \left\{ s \in \Sigma^* \left| \widehat{\delta}(q_0, s)! \right. \right\} \tag{37}$$

$$L_m(G) := \left\{ s \in \Sigma^* \left| \widehat{\delta}(q_0, s) \in Q_m \right. \right\} \tag{38}$$

3.5.2 REPRESENTAÇÃO GRÁFICA DE SEDTS

A representação gráfica do modelo de um SEDT consiste de dois elementos: um modelo das atividades, não temporizado, e o autômato de estados finitos, com os respectivos limites de tempo. O modelo das atividades é representado por um gráfico de transições, chamado de ATG - *activity transition graph*, que descreve o comportamento não temporizado do sistema, e o TTG - *timed transition graph*, que incorpora o evento *tick*. Na Figura 14 pode-se observar exemplos de ATGs.

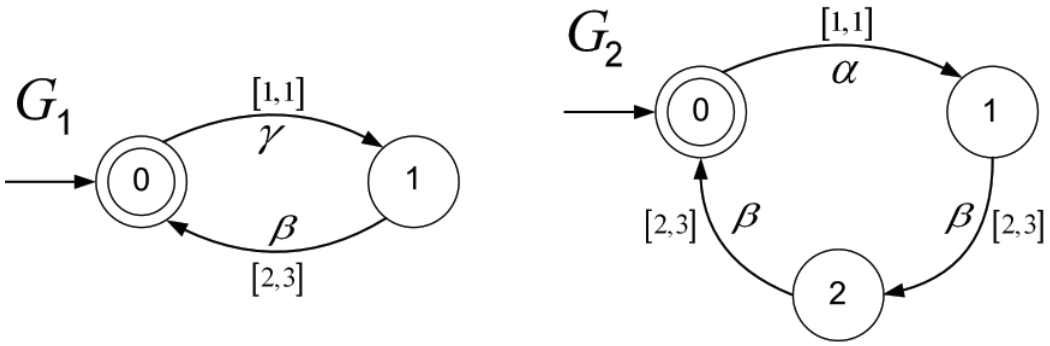

Figura 14 – Exemplos de $ATGs$.

Para a planta G_1 da Figura 14, temos:

$$G_{1act} = (\Sigma_{1act}, Q, \delta, q_0, Q_m) \tag{39}$$

onde:

$$\begin{array}{ll} \Sigma_{1act} = (\gamma, \beta, tick) & \sigma = \{\gamma, 1, 1\}, \{\beta, 2, 3\} \\ q_0 = a_0 = 0 & Q_m = \{0\} \\ \delta_{act}(\gamma, 0) = 1 & \delta_{act}(\beta, 1) = 0 \end{array}$$

O intervalo associado ao evento γ é $[1,1]$, e ao evento β é $[2,3]$.

Para o evento γ, em uma unidade de tempo (uma ocorrência do evento $tick$), a partir do estado inicial, o mesmo estará habilitado e, como o tempo superior foi também alcançado, γ deve ocorrer obrigatoriamente antes de qualquer outro evento $tick$. O evento β pode ocorrer após os primeiros 2 $ticks$, ou sempre que 2 $ticks$ tiverem ocorrido após o último γ, e deve ocorrer ao menos uma vez a cada 3 $ticks$. Na Figura 15 pode-se observar os TTGs para as plantas G_1 e G_2.

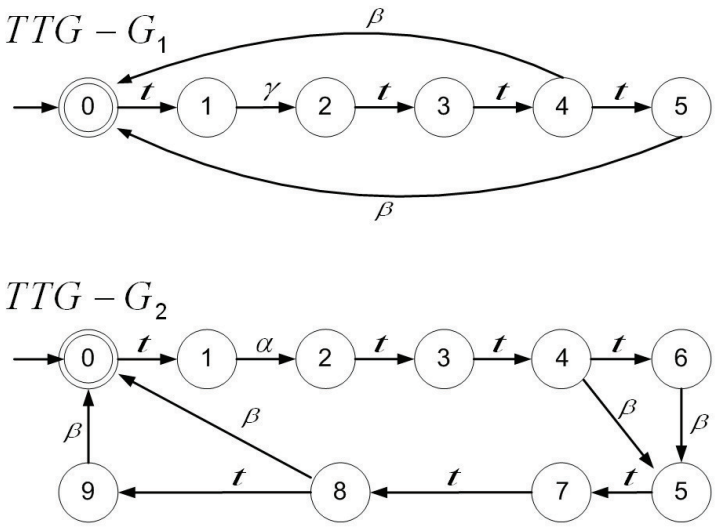

Figura 15 – TTG **para as plantas da figura 14.**

3.5.3 OPERAÇÕES SOBRE AUTÔMATOS TEMPORIZADOS

Composição síncrona: Sejam G_1 e G_2, dois SEDTs, com os alfabetos Σ_1 e Σ_2, respectivamente, onde $\Sigma_i = \Sigma_{iact} \cup tick$. Para formar a composição de G_1 e G_2, primeiramente se deve definir o alfabeto de G como $\Sigma_1 \cup \Sigma_2$ e a estrutura *ATG* de G como o produto síncrono de ambos os ATGs:

$$G_{act} = G_{1act} \| G_{2act} \tag{40}$$

Para cada evento $\sigma \notin (\Sigma_{1act} \cap \Sigma_{2act})$ os limites de tempo (l_σ, u_σ) permanecem inalterados enquanto que, se $\sigma \in (\Sigma_{1act} \cap \Sigma_{2act})$, os limites de tempo em G são definidos por:

$$(l_\sigma, u_\sigma) = (\max(l_{1,\sigma}, l_{2,\sigma}), \min(u_{1,\sigma}, u_{2,\sigma})) \tag{41}$$

com $l_\sigma \leq u_\sigma$, e caso esta condição não seja satisfeita, a composição G é considerada indefinida. A componente com o maior valor para o limite inferior (e com o menor valor para o limite superior) determina o comportamento temporizado da composição G.

Para realizar a composição de modelos de plantas em SEDTs, é utilizada a função *Comp* do *software* TTCT. Essa função apresenta o resultado de uma composição, considerando os limites e condições definidos por 41 e 38.

A função *Comp* traz um resultado muito diferente da composição síncrona entre as estruturas temporizadas (TTGs) de duas plantas. Isso porque, ao se obter o TTG global da planta, a partir do TTG de cada subsistema, a sincronização do evento *tick* pode gerar inconsistências irreais no resultado, podendo até levar a um *deadlock*, ou bloqueio no sistema (WONHAM, 2011). Situação que pode ser observada na Figura 16, da composição síncrona das plantas G_1 e G_2 da Figura 14.

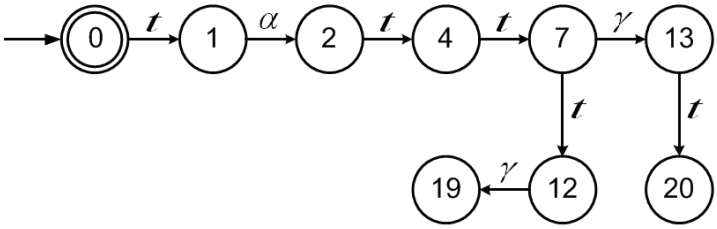

Figura 16 – Situação de bloqueio na composição síncrona $TTG\_G1 \| TTG\_G2$.

O resultado correto, obtido por meio da função $Comp(G_1, G_2)$, mostrado na figura 17, é livre de situações de bloqueio:

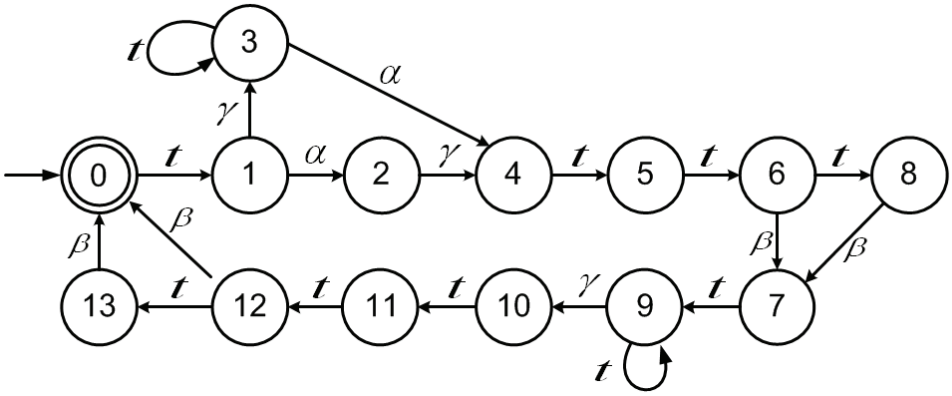

Figura 17 – Resultado da operação $Comp(G_1, G_2)$.

3.5.4 CONTROLE SUPERVISÓRIO DE SEDTS

Analogamente ao controle supervisório em SED, nos sistemas temporizados também deve-se definir os chamados *eventos controláveis*, transições que podem ser desabilitadas ou *proibidas* de ocorrer. Dentro dos eventos remotos de um SEDT, um novo subconjunto $\Sigma_{hib} \subseteq \Sigma_{rem}$ de eventos *proibitíveis* deve ser definido, onde o supervisor deve apagar um evento proibido da lista das transições que podem ocorrer em determinado estado q do SEDT supervisionado.

Outra categoria de eventos que deve também ser definida em um modelo temporizado diz respeito aos eventos *forçáveis*, $\Sigma_{for} \subseteq \Sigma_{act}$. Eventos forçáveis são os eventos que podem ser desabilitados ou habilitados (forçados) por um supervisor externo. O conjunto de eventos controláveis Σ_c representa a união dos eventos proibitíveis Σ_{hib} com os forçáveis Σ_{for}, $\Sigma_c = \Sigma_{hib} \cup \Sigma_{for}$, e o conjunto de todos os eventos, nessa abordagem é dada por $\Sigma = \Sigma_c \cup \Sigma_{unc} \cup tick$. Um evento em Σ_{rem} pode ser forçável e proibitível.

O conjunto de eventos *incontroláveis* é definido por:

$$\Sigma_{unc} = \Sigma_{act} - \Sigma_{hib} = \Sigma_{spe} \cup (\Sigma_{rem} - \Sigma_{hib}) \tag{42}$$

e o conjunto dos eventos *controláveis* é dado por:

$$\Sigma_c = \Sigma - \Sigma_{unc} = \Sigma_{hib} \cup \{tick\} \tag{43}$$

Observar que um evento forçável pode ser controlável ou incontrolável; e a ocorrência de um evento forçável que é incontrolável não pode ser desabilitada. Por exemplo, um controle de tráfego aéreo pode forçar a aterrissagem de uma aeronave dentro de um determinado intervalo de tempo, mas não pode evitar que a mesma execute a operação. A aterrissagem é forçável, mas não controlável (BRANDIN; WONHAM, 1994).

Seja considerado um gráfico das atividades de uma planta G, onde cada nó n representa uma cadeia s da linguagem gerada por G, $L(G)$. Em n pode ser definido o subconjunto de eventos elegíveis $Elig_G(s) \subseteq \Sigma$:

$$Elig_G(s) := \{\sigma \in \Sigma \,|\, s\sigma \in L(G)\} \tag{44}$$

e $Elig_G(s) \neq 0$ para todo $s \in L(G)$. Um supervisor é um tomador de decisão que, em n, seleciona um subconjunto não vazio de $Elig_G(s)$.

Considerando um supervisor V, e V/G para denotar o par (G,V), ou G sob supervisão de V, e também V é dito não bloqueante para G, quais:

$$\overline{L_m(V/G)} = L(V/G) \tag{45}$$

Seja $K \subseteq L(G)$, e:

$$Elig_K(s) := \left\{\sigma \in \Sigma \,\middle|\, s\sigma \in \overline{K}\right\}, s \in \Sigma^* \tag{46}$$

a ocorrência de um evento não-controlável σ em $L(G)$, após uma cadeia s, é tal que mantém o comportamento de G na especificação K, $s\sigma \in \overline{K}$.

K é dito controlável e.r.a G se, para todo $s \in \overline{K}$:

$$Elig_K(s) \supseteq \begin{cases} Elig_G(s) \cap (\Sigma_{unc} \cup \{tick\}) & \text{, se } Elig_K(s) \cap \Sigma_{for} = \emptyset \\ Elig_G(s) \cap \Sigma_{unc} & \text{, se } Elig_K(s) \cap \Sigma_{for} \neq \emptyset \end{cases} \tag{47}$$

ou seja, K controlável significa que um evento σ (no alfabeto Σ incluindo o evento $tick$), pode ocorrer em K se:

i) não for controlável ou for $tick$ e elegível em G, e não há nenhum evento forçável elegível em K, ou;

ii) não for controlável em G, mas um evento forçável é elegível em K.

Pode-se afirmar que na ocorrência de um evento $tick$ que resulte fora da especificação K, o mesmo pode ser precedido por um evento forçável dentro de K. Dois mecanismos de controle são previstos: a prevenção de cadeias proibidas a partir da desabilitação de eventos, e os eventos forçáveis.

Em (BRANDIN; WONHAM, 1993) é demonstrada a existência de uma máxima sublinguagem controlável K para SEDTs. Tal linguagem é dada por $SupC(K,G)$, com um supervisor S tal que $L_m(S/G) = SupC(K,G)$, representando uma supervisão ótima com o comportamento da planta G sendo minimamente restringido, de modo que a especificação global K seja atendida.

3.5.5 CONTROLE SUPERVISÓRIO MODULAR DE SEDTS

Uma das definições necessárias para o cálculo de supervisores modulares diz respeito à conjunção de supervisores. Sejam considerados então, dois supervisores S_1 e S_2, de uma planta G:

i) S_1 e S_2 são $trim$;

ii) $L_m(S_1)$ e $L_m(S_2)$ são controláveis e.r.a G;

iii) S_1/G e S_2/G são não bloqueantes, $\overline{L_m(S_1/G)} = L(S_1/G), \overline{L_m(S_2/G)} = L(S_2/G)$.

A função de conjunção entre S_1 e S_2, denotada por $S_1 \wedge S_2$, é a habilitação de um evento σ, se e somente se, σ for habilitada simultaneamente por S_1 e S_2. Para uma melhor definição, três teoremas são apresentados (BRANDIN; WONHAM; BENHABIB, 1992):

Teorema 1: $S_1 \wedge S_2$ é um supervisor de G se, e somente se:

i) $S_1 \wedge S_2$ for $trim$;

ii) $L_m(S_1/G)$ e $L_m(S_2/G)$ forem não-conflitantes, i.e.:

$$\overline{L_m(S_1/G) \cap L_m(S_2/G)} = \overline{L_m(S_1/G)} \cap \overline{L_m(S_2/G)} \tag{48}$$

A condição da equação 48 é testada através do comando *NonConflict* do *TTCT*:

$$NonConflict\left(L_m(S_1/G), L_m(S_2/G)\right) \tag{49}$$

iii) $L(S_1)$ e $L(S_2)$ forem conjuntamente coercivos.

Definição: Coercividade

Seja $K \subseteq L(G)$. K é dito $coercivo$ e.r.a G, se:

$$\left(\forall s \in \overline{K}\right) tick \in Elig_G(s) - Elig_K(s) \Rightarrow Elig_K(s) \cap \Sigma_{for} \neq \emptyset \tag{50}$$

isto é,

$$\left(\forall s \in \overline{K}\right) Elig_K(s) \cap \Sigma_{for} = \emptyset \quad \& \quad Elig_G(s) \Rightarrow tick \in Elig_K(s) \tag{51}$$

$K_1, K_2 \subseteq L(G)$ são ditas conjuntamente coercivos em relação a G, se $K_1 \cap K_2$ for coercivo em relação a G.

Assim, se dois supervisores S_1 e S_2 atenderem às condições do Teorema 1, os mesmos são considerados apropriados para o controle da planta G (BRANDIN; WONHAM, 1993).

Nesse capítulo foi apresentada alguns conceitos da teoria de linguagens regulares e autômatos determinísticos de estados finitos nas quais a TCS, segundo a abordagem de Ramadge e Wonham, é baseada. Algumas das extensões da TCS, como supervisores modulares e controle supervisório para SEDTs também foram apresentadas.

Nos próximos capítulos são apresentados os conceitos dos problemas de escalonamento na indústria de manufatura, e a proposta para a resolução de tais problemas por meio da aplicação da Teoria de Controle Supervisório e de suas extensões para modelos temporizados.

4 ESCALONAMENTO DA PRODUÇÃO

Na história da evolução fabril, uma dos grandes marcos foi o advento da indústria mecanizada, iniciada com a Revolução Industrial, em meados do século XVIII na Inglaterra (ASHTON, 1997). As unidades de produção e células de manufatura, desde então, aumentaram muito em complexidade, sendo compostas por várias máquinas interligadas por sistemas automatizados de transporte e manuseio de peças. Porém nem sempre a utilização dos recursos disponíveis em linhas automatizadas é feita da melhor maneira possível. Os desafios de um mercado cada vez mais competitivo fazem com que engenheiros e gerentes de plantas industriais procurem métodos eficientes no controle das atividades nessas células.

Muitas indústrias contam com sistemas totalmente informatizados do tipo *Manufacturing Resource Planning* (MRP) de planejamento da produção. Tais ferramentas consideram prognósticos de vendas, de preços e de demandas de produtos, auxiliam no controle de estoque e no fluxo das ordens de fabricação e de matéria prima entre as diversas células de manufatura. Mas, apesar de proporcionar relatórios com riqueza de detalhes e bastante precisão, e conseguir que todas as peças seja entregues na data prevista, ainda assim, não se garante que as máquinas estejam operando de uma maneira eficiente, sem qualquer desperdício de tempo. Com sistemas MRP, não é possível, por exemplo, avaliar capacidades produtivas instaladas (RODAMMER; WHITE, 1988). Como exemplos comerciais de tais sistemas podem ser citados SAP, Oracle, JDEdwards, Baan, Peoplesoft, dentre outros (SACCOL et al., 2004).

O escalonamento de tarefas, por sua vez, define o sequenciamento de atividades e a determinação dos tempos de duração de cada processo. Pode também detectar conflitos de recursos, assegurar que a matéria prima seja solicitada em tempo hábil para atender à linha de produção, e pode identificar períodos de tempo disponíveis para que manutenções preventivas de instalações e equipamentos sejam feitas (HERRMANN, 2007).

O objetivo desse capítulo, é apresentar alguns conceitos básicos de escalonamento, algumas das abordagens encontradas na literatura, e propor um método para a obtenção de um escalonamento ótimo de tarefas baseada nos conceitos da teoria de controle supervisório (RAMADGE; WONHAM, 1989).

4.1 PROBLEMAS DE ESCALONAMENTO

Um sistema de manufatura flexível proporciona a possibilidade de produção em grandes volumes, com a versatilidade que normalmente somente as células isoladas de trabalho possuem. Porém, devido à quantidade de máquinas e operações envolvidas, o escalonamento torna-se uma tarefa bastante complexa.

Formalmente, um problema de escalonamento pode ser definido como um conjunto finito $J = \{J_1, J_2, \ldots, J_n\}$ de n tarefas ou *jobs*, que devem ser executados em um conjunto finito $M = \{M_1, M_2, \ldots, M_m\}$ de m máquinas. Cada tarefa J_i consiste de uma sequência de operações $O = O_{i1}, O_{i2}, ..., O_{im}$, as quais devem seguir uma determinada ordem, definida pelas especificações de precedência de cada uma das operações. Cada operação O_{im} representa uma operação da tarefa J_i, deve ser processada na máquina M_m , por um período de tempo ininterrupto,*i.e.*, nenhuma operação ou tarefa pode ser interrompida (JAIN; MEERAN, 1999).

Cada operação pode ser caracterizada por uma notação do tipo (M_m, d) com $M_m \in M$ e $d \in \mathbb{N}$, indicando a utilização da máquina M_m por um período de tempo d (ABDEDDAIM; ASARIN; MALER, 2006), (JAIN; MEERAN, 1999). O objetivo é então, determinar o tempo de início de cada operação, para se minimizar o tempo de execução total de cada tarefa, ou seja, definir o *schedule*.

Além de definir a ordem na qual um recurso irá executar determinada tarefa, o escalonamento deve também atender critérios de desempenho como, por exemplo, prazo de entrega, quantidade máxima de produção ou mínimo tempo de execução nas tarefas.

Outras considerações a serem feitas, no desenvolvimento de um escalonamento, dizem respeito ao modo de execução das tarefas (FRENCH, 1982):

- preempção não permitida, isto é, uma vez iniciada, uma operação em uma máquina deve ser finalizada antes que outra operação inicie na mesma máquina;
- cada tarefa possui m operações distintas, uma em cada máquina. Uma tarefa não exige que sejam realizadas duas operações na mesma máquina; e
- nenhuma máquina executa duas operações simultaneamente.

Um dos primeiros algoritmos para resolver um problema de escalonamento, foi desenvolvido na década de 50 (JOHNSON, 1954), para uma célula composta por duas máquinas e que minimiza o tempo máximo de fluxo de operações.

French (1982) demonstra que, para uma célula com n tarefas e m máquinas, existem $(n!)^m$ soluções. Um problema do tipo 20×10 possui, portanto, $(20!)^{10} = 72.651 \times 10^{183}$ soluções possíveis. Mesmo que algumas das soluções propostas não sejam factíveis devido a situações de precedência ou interrupção, a enumeração de todas as possibilidades é impraticável, e os problemas de escalonamento são classificados como *NP-Hard*, indicando a complexidade matemática envolvida na sua solução (PINHA, 2010), (JAIN; MEERAN, 1999), (ABDEDDAIM; ASARIN; MALER, 2006).

4.1.1 DEFINIÇÃO DE ESCALONAMENTO

Para que se possa definir um escalonamento ou *schedule* de máquinas, sejam considerados (ABDEDDAIM; ASARIN; MALER, 2006):

- um conjunto $P = \{p_1, ..., p_m\}$ de operações;
- um conjunto $M = \{m_1, ..., m_n\}$ de máquinas;
- uma função $\mu : P \to M$ de atribuição de operações às máquinas;
- uma função $st : P \to \mathbb{R}_+$, que indica o tempo de início de cada operação ($\mathbb{R}_+$ é o conjunto dos números reais positivos);
- uma função duração de cada operação definida por $d : P \to \mathbb{N}$.

Deseja-se encontrar o escalonamento que minimize o tempo total de execução das operações, respeitando-se as premissas citadas de modo de execução das tarefas, na seção 4.1.

Considerando-se que todas as máquinas são distintas, e que cada operação p pode ser executada somente em uma máquina $\mu(p)$, o final da operação en é dado então por $en(p) = st(p) + d(p)$.

Seja a operação predecessora imediata de p, p' denotado por $p' \prec p$, um escalonamento é factível se as seguintes condições forem satisfeitas:

Precedência: para todo $p, p' \in P, p' \prec p \Rightarrow en(p) \leq st(p')$;

Exclusão mútua: para todo $p, p' \in P$ e $\mu(p) = \mu(p')$, tal que:

$$\{st(p), en(p)\} \cap \{st(p'), en(p')\} = \emptyset \tag{52}$$

O tamanho do escalonamento é dado por $D = \max\{en(p) : p \in P\}$, e o escalonamento ótimo é o escalonamento factível com o tamanho mínimo, $min(D)$ (ABDEDDAIM; ASARIN; MALER, 2006).

4.1.2 CLASSIFICAÇÃO DE PROBLEMAS DE ESCALONAMENTO

A notação utilizada para fazer a classificação dos problemas de escalonamento, conforme proposta por (FRENCH, 1982), é feita por quatro caracteres separados por uma barra:

$$n/m/A/B \tag{53}$$

onde:

n - número de tarefas;

m - número de máquinas;

A - fluxo do processo;

B - critério de desempenho.

O fluxo do processo é definido pelo tipo de problema a ser tratado e quantidade de máquinas consideradas: se $m = 1$, somente um estágio na célula e $A = \{\}$. Se $m \neq 1$ então A pode assumir os valores:

F - problemas do tipo *flowshop*, ou de programação na produção;

J - problemas do tipo *jobshop*, ou de escalonamento;

G - problemas do tipo *general jobshop*, onde não há restrição nenhuma na execução das tarefas.

O parâmetro B define o critério pelo qual o escalonamento será avaliado, por exemplo, tempo de espera mínimo, tempo médio de espera, atraso médio ou tempo de fluxo mínimo.

4.1.3 MÉTODOS UTILIZADOS DE ESCALONAMENTO

Além dos critérios mencionados, que atendem às necessidades de produção, requisitos de um controle supervisório de atividades nas diversas células de manufatura devem ser considerados. Sendo possível com isso, evitar situações de bloqueio e de falta ou excesso de peças em armazéns intermediários de processos (*buffers*) (GOLMAKANI; MILLS; BENHABIB, 2003).

Conforme French (1982), as indústrias buscam uma sequência onde as tarefas são executadas pelas máquinas, de modo que:

- sejam compatíveis com as restrições tecnológicas, *i.e.*, sejam factíveis; e
- sejam ótimas com relação a algum critério de desempenho.

Esses critérios podem ainda ser divididos em (SLACK et al., 2009):

- qualidade na produção;
- velocidade de manufatura de um item;
- entrega no prazo definido pelos clientes;
- flexibilização na linha de manufatura; e
- custo de produção.

Tais necessidades motivaram um grande número de pesquisadores a desenvolver várias abordagens para o modelamento e solução de problemas de escalonamento, incluindo sistemas híbridos, lógica *Fuzzy*, técnicas de inteligência artificial e algoritmos genéticos.

Lógica *Fuzzy*: a utilização de lógica *Fuzzy* (ZADEH, 1965) na solução de problemas de escalonamento é considerada, devido à sua facilidade de tratamento de informações incompletas, e também pelo fato de que os conhecimentos de especialistas podem ser facilmente codificadas em regras *Fuzzy* (NANVALA, 2011). Bilkay propõe o desenvolvimento de um algoritmo de decisão baseado em lógica difusa, para determinar a prioridade de processamento de peças em máquinas, considerando tamanho de lote, prazo de entrega, tempo total de processamento e ferramentais necessários (BILKAY; ANLAGAN; KILIC, 2004).

Sistemas híbridos: uma abordagem híbrida utilizando redes neurais adaptativas e heurísticas é proposta por Yang (2000), onde a estratégia é utilizada para a solução de problemas de escalonamento. Duas heurísticas são combinadas para acelerar e assegurar a convergência de uma rede neural. Os pesos sinápticos e os limiares de ativação (*biases*) das conexões da rede neural são ajustados de acordo com o sequenciamento e restrições de recursos, durante o processamento de um escalonamento (YANG; WANG, 2000). Rooda (1994) traduz o problema de escalonamento, com suas restrições e sequências de tarefas pré-definidas, em um modelo de programação linear inteiro, que facilita o seu mapeamento em uma estrutura adequada de uma rede neural artificial. A rede é responsável então por encontrar a relação de prioridade entre as diversas tarefas e determinadas máquinas (WILLEMS; ROODA, 1994).

Redes neurais artificiais: Terra (2000) faz o uso de simulação computacional de sistemas fabris e redes neurais artificiais. Um modelo, com todos os parâmetros de uma célula de manufatura é construído e, ao final da simulação, alguns valores de medidas de desempenho são coletados. Nessa abordagem, a rede neural artificial deve aprender as regras de precedência designadas às máquinas, e os valores de medidas de desempenho pré-estabelecidos, de modo que se encontre a melhor alternativa na programação da produção (TERRA; PEREIRA, 2000).

Jones (1998) porém, afirma que algoritmos neurais consomem muito tempo no processo de convergência e, além disso, sua manutenção é difícil, uma vez que estão relacionados diretamente ao sistema para o qual foram desenvolvidos, ou seja, a generalização não é possível (JONES; RABELO, 1998). Ajustes eventualmente necessários devem ser feitos por especialistas, tornando as técnicas complexas para os usuários, e também apresentam pouca fidelidade com sistemas reais, dificultando sua implementação em plantas industriais (PINHA, 2010).

Algoritmos genéticos: Falkenauer et al (1991) utiliza um algoritmo genético para a solução de um problema de escalonamento com prazo de entrega, apresentando um operador de cruzamento (*crossover*), chamado *Linear Order Crossover* (OLX). O operador OLX, de acordo com os autores, preserva informações de alta e baixa prioridade das operações no processo de cruzamento,*i.e.* o início e o fim dos cromossomos não são perdidos. Silva (2011) propõe o uso de algoritmos clonais, que fazem a busca aleatória sobre as permutações das sequências de operação. Com o uso da TCS, a busca é conduzida somente sobre o conjunto de soluções possíveis do problema proposto (SILVA, 2011).

Um aspecto a ser observado, diz respeito à modularidade, presente no método proposto para a solução de problemas de escalonamento, por meio da TCS e autômatos temporizados. Em outras abordagens onde há a necessidade de um grande número de equações, aspectos da composição de subsistemas são menos explícitos (ABDEDDAIM; ASARIN; MALER, 2006).

A principal vantagem de uma abordagem baseada em autômatos, é a possibilidade de obtenção de um controle supervisório de uma maneira automática, com o uso de ferramentas computacionais. Essas ferramentas garantem a precisão dos cálculos e asseguram a obtenção de um controle não-bloqueante e minimamente restritivo (BRANDIN; WONHAM; BENHABIB, 1992), (RAMADGE; WONHAM, 1989).

5 MÉTODO PARA O ESCALONAMENTO DE TAREFAS POR INTERMÉDIO DA TCS

Do ponto de vista de controle, é desejável que em qualquer sistema, uma situação de bloqueio seja evitada. Isso pode ser garantido com a teoria de controle supervisório (RAMADGE; WONHAM, 1989), descrita no capítulo 3. Porém, considerando critérios de desempenho, tendo como indicador a variável *tempo*, um comportamento não-ótimo pode estar presente no sistema sob controle. Como todas as situações de bloqueio são evitadas, existem diversas sequências de ações que podem levar ao mesmo resultado, que nem sempre implicam em desempenhos similares, e algumas cadeias de eventos podem ser mais rápidas que outras.

Nesta seção será mostrado um método para a obtenção de escalonamento em células de manufatura, com a utilização de modelos de autômatos temporizados e conceitos da teoria de controle supervisório.

Tal método é composto pelas seguintes etapas:

1. Descrição de operação e modelagem da planta;
2. Modelagem das especificações;
3. Síntese do supervisor monolítico temporizado e obtenção do escalonamento;
4. Síntese do supervisor modular temporizado e obtenção do escalonamento.

5.1 DESCRIÇÃO DE OPERAÇÃO E MODELAGEM DA CÉLULA DE MANUFATURA

A célula de manufatura considerada possui duas máquinas M_1 e M_2, um *buffer* B de capacidade unitária e duas esteiras de capacidades infinitas de entrada e de saída E_1 e E_2.

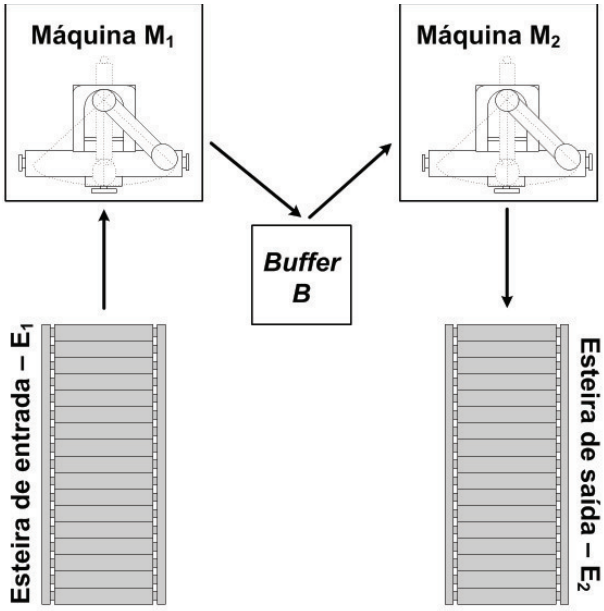

Figura 18 – Célula de manufatura com duas máquinas e duas esteiras.

Trata-se uma linha de transferência, onde a máquina M_1 recebe uma peça da esteira E_1, realiza uma operação sobre a mesma e, ao finalizar a operação, descarrega automaticamente em B, estando consequentemente disponível para a máquina M_2. A máquina M_2 realiza o processamento e entrega para a esteira de saída E_2. Uma vez um ciclo tenha iniciado, as máquinas podem finalizar a operação ou então, uma falha pode ocorrer, sendo neste caso, reparadas imediatamente.

Como mencionado, as máquinas podem estar ambas disponíveis ou não para produção, e seus estados e eventos podem ser observados na figura 19 do diagrama ATG de $M1$ e $M2$:

- I_i - máquina M_i está ociosa;
- W_i - máquina M_i em operação;
- D_i - máquina M_i danificada;
- R_i - máquina M_i em reparo;
- α_i - iniciar operação;
- β_i - finalizar operação;
- λ_i - falha;
- μ_i - reparo inicializado;
- η_i - reparo finalizado.

Sendo o conjunto dos eventos não controláveis formado por $\Sigma_{unc} = \{\beta_1, \lambda_1, \eta_1, \beta_2, \lambda_2, \eta_2\}$ e o conjunto dos eventos forçáveis formados por $\Sigma_{for} = \{\alpha_1, \mu_1, \alpha_2, \mu_2\}$.

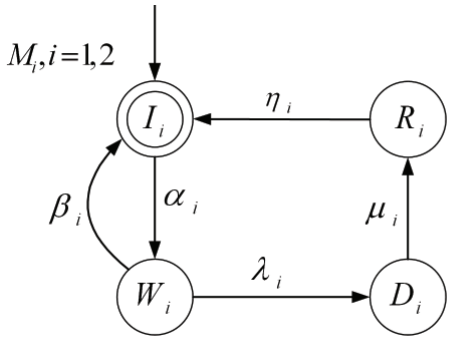

Figura 19 – Modelagem ATG das máquinas M_1 e M_2.

Os limites de tempo inferior l_σ e superior u_σ para cada operação são mostrados na Tabela 1:

Tabela 1 – Limites de tempos de processamento.

		Limites de tempo	
Máquina	Operação	l_σ	u_σ
M_1	α_1	0	∞
	β_1	1	2
	λ_1	0	2
	μ_1	0	∞
	η_1	1	∞
M_2	α_2	0	∞
	β_2	1	1
	λ_2	0	1
	μ_2	0	∞
	η_2	2	∞

As Figuras 20 e 21 mostram os gráficos TTG para as máquinas M_1 e M_2. A estrutura lógica é mantida, porém pode-se observar a complexidade gerada, com a inclusão do evento do relógio $tick$, representado por t na estrutura de transição dos gráficos $TTGs$.

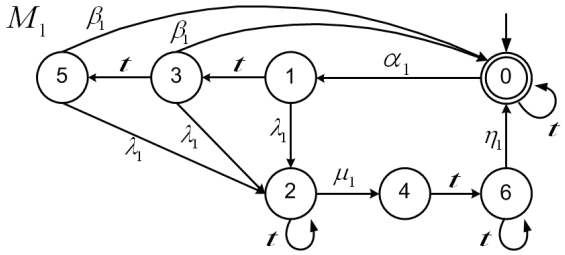

Figura 20 – Modelagem TTG da máquina M_1.

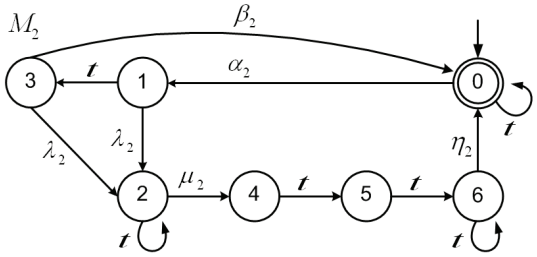

Figura 21 – Modelagem TTG da máquina M_2.

5.2 MODELAGEM DAS ESPECIFICAÇÕES

O comportamento em malha fechada da célula de manufatura deve satisfazer às seguintes condições:

i) não pode haver *overflow* no *buffer*, ou seja, a máquina M_1 não pode descarregar uma peça em B, se o mesmo estiver cheio;

ii) não pode haver *underflow* no *buffer*, ou seja, M_2 não pode iniciar processo de retirada de peça, caso B esteja vazio;

iii) caso ocorra falha em ambas as máquinas (eventos λ_1 e λ_2), o reparo de M_2 (evento μ_2) deve ser inicializado antes do reparo de M_1 (evento μ_1);

As duas primeiras restrições são implementadas com o próprio autômato TTG do *buffer* B, conforme a figura 22.

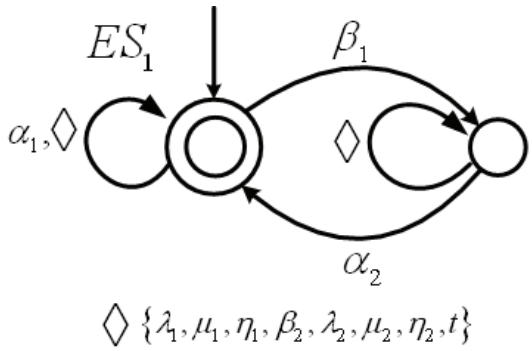

Figura 22 – Modelagem das condições de não *overflow* e não *underflow*.

A condição de reparo é implementada com o autômato TTG da figura 23;

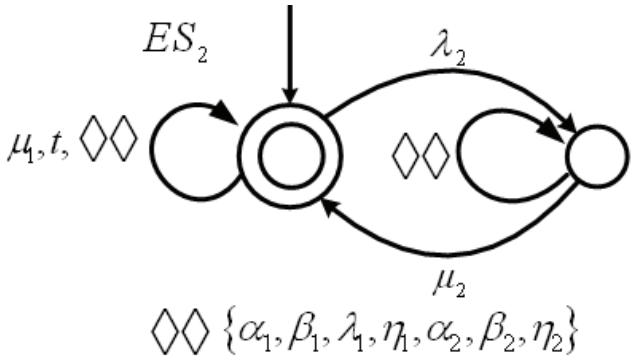

Figura 23 – Modelagem da condição de reparo.

5.3 ABORDAGEM MONOLÍTICA E CÁLCULO DO ESCALONAMENTO

O comportamento em malha fechada da célula, de tal modo que satisfaça às especificações lógicas impostas, é obtido conforme descrito a seguir:

i) Com os autômatos ATG das máquinas M_1 e M_2 modelados, pode-se obter a planta M, com a composição paralela de M_1 e M_2, que descreve o comportamento em malha aberta do sistema. Esse cálculo é feito por meio do comando $Comp$ do aplicativo $TTCT$, mencionado no capítulo 3. Obtém-se então um SEDT com 49 estados e 134 transições. No presente trabalho, a quantidade de estados e transições é apresentada após o cálculo, entre parênteses.

$$M = Comp\,(M_1, M_2)\,;(49 \text{ estados, } 134 \text{ transições}) \tag{54}$$

ii) Para a obtenção da combinação lógica das especificações, é utilizado o comando $Meet$ da ferramenta $TTCT$, conforme descrito em 3.2.3. Assim, a especificação lógica completa é dada pela combinação das especificações $ES1$ e $ES2$, formando um SEDT com 4 estados e 30 transições:

$$ES = Meet\,(ES_1, ES_2)\,;(4 \text{ estados, } 30 \text{ transições}) \tag{55}$$

iii) O comportamento em malha fechada que atenda às especificações, e que seja o menos restritivo possível é dado pelo comando $Supcon$:

$$SUP = Supcon\,(M, ES)\,;(48 \text{ estados, } 105 \text{ transições}) \tag{56}$$

SUP com 48 estados e 105 transições é a linguagem suprema de controle da planta representada por M, que atende às especificações lógicas representadas por ES.

iv) Para atender à especificação de prazo, Brandin e Wonham (1992) consideram um temporizador que representa uma sequência onde todos os estados são marcados, e a transição ocorre somente com a ocorrência do evento $tick$. Para uma especificação de prazo igual a 10 unidades de tempo, esse temporizador, sincronizado por meio do comando $meet$ com um SEDT, faz com que o sistema execute todas as tarefas em no máximo 10 *ticks*, *i.e.*, a partir do $11^{\underline{o}}$ evento, somente o evento do relógio está habilitado. Um *loop* com todos os eventos, *i.e.* $\Sigma_c \cup \Sigma_u$, menos o evento $tick$ está também presente em cada um dos estados, com exceção do último, que possui um *loop* somente com o evento $tick$.

O temporizador descrito pode ser observado na figura 24.

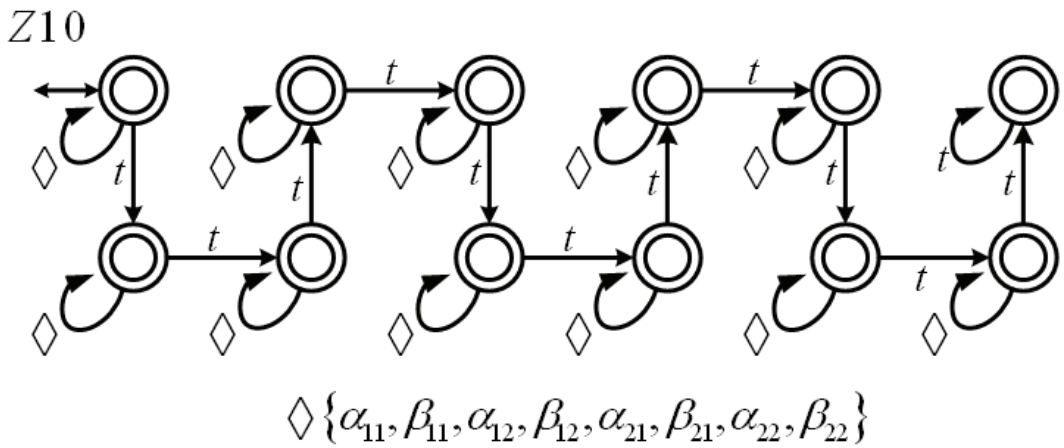

Figura 24 – Temporizador.

Por intermédio da sincronização do comportamento de $Z10$ mostrado na figura 24 com o comportamento da planta M, obtém-se a sequência que corresponde a um tempo de ciclo de produção com uma duração máxima igual ao tamanho do temporizador. Esse procedimento mostra as cadeias que satisfazem a restrição imposta, ou seja, as tarefas que podem ser executadas em no máximo 10 unidades de tempo. Para determinar se tal garantia é factível, é suficiente verificar se a suprema sublinguagem controlável $TSUP$ é diferente de $vazio$, utilizando-se o comando $Supcon$:

$$TSUP = Supcon\,(SUP, Z10)\,;(90 \text{ estados, } 134 \text{ transições}) \tag{57}$$

Para o exemplo, obteve-se um SEDT com 90 estados e 134 transições, comprovando que o resultado é possível de ser obtido, *i.e.*, existe uma cadeia que atende à restrição temporal imposta.

5.4 ABORDAGEM MODULAR E CÁLCULO DO ESCALONAMENTO

Na seção anterior foi apresentada uma abordagem monolítica, segundo Wonham (1993), para a obtenção de uma suprema linguagem controlável, que atenda à restrição de tempo mínimo de ciclo.

Na abordagem modular, a tarefa do supervisor é dividida em duas ou mais subtarefas. Além de ser obtida mais facilmente, a manutenção de um supervisor modular é também mais simples. Caso uma tarefa deva ser novamente especificada, basta a modificação do supervisor correspondente, trazendo maior flexibilidade quando comparado com a abordagem monolítica (BRANDIN; WONHAM; BENHABIB, 1992).

Tomando-se novamente o exemplo utilizado na seção 5.4, a abordagem modular é obtida conforme segue:

i) Da mesma forma como obtido na equação (55), M é a composição paralela de M_1 e M_2 resultando em um SEDT com 49 estados e 134 transições.

ii) Sejam as especificações ES_1 e ES_2. O comando $SupCon(M, ES_1)$ e $SupCon(M, ES_2)$ reconhece a condição $Trim$ de $Lm(ES_1)$ e $Lm(ES_2)$, respectivamente, em relação a M (WONHAM, 2011):

$$SupES_1 = Supcon(M, ES_1); (51 \text{ estados}, 123 \text{ transições}) \tag{58}$$

$$SupES_2 = Supcon(M, ES_2); (48 \text{ estados}, 121 \text{ transições}) \tag{59}$$

Para verificar a condição de não-conflito entre os supervisores modulares, utiliza-se o comando $NonConflict$ do $TTCT$:

$$true = NonConflict(SupES_1, SupES_2) \tag{60}$$

O comando $Condat$ do aplicativo $TTCT$:

$$DAT = condat(M, Sup) \tag{61}$$

retorna os dados de controle DAT para o supervisor Sup da planta controlada M.

Caso Sup tenha sido calculado previamente por meio do comando $Supcon$, então DAT irá tabular os eventos que devem ser desabilitados em cada estado de Sup. $Condat$ pode ser utilizado para verificar se a linguagem representada por Sup é controlável em relação a M, bastando para isso analisar se os eventos desabilitados em DAT são controláveis (WONHAM, 2011).

Fazendo o cálculo para $SupES_1$ e $SupES_2$, obtém-se:

$$SupES_1DAT = Condat(M, SupES_1); \text{controlável} \tag{62}$$

$$SupES_2DAT = Condat(M, SupES_2); \text{controlável} \tag{63}$$

Satisfeitas as condições do Teorema 1, então $SupES_1$ e $SupES_2$ são supervisórios apropriados para M (WONHAM, 2011).

iii) A especificação de tempo é atendida, conforme visto na obtenção do supervisor monolítico, com o temporizador $Z10$. O respectivo supervisor é obtido com o comando $Supcon$:

$$SupZ10 = Supcon(M, Z10); (111 \text{ estados, } 209 \text{ transições}) \tag{64}$$

Fazendo a verificação com relação ao Teorema 1, testando-se a condição de não-conflito entre os supervisores modulares, e a condição de controlabilidade em relação à planta M:

$$true = NonConflict(SupES_1, SupZ10) \tag{65}$$

$$true = NonConflict(SupES_2, SupZ10) \tag{66}$$

$$SupZ10DAT = Condat(M, SupZ10); \text{controlável} \tag{67}$$

E $SupZ10$ é um supervisor apropriado para M, *i.e.* atende ao Teorema 1.

O supervisor modular R é obtido com a conjunção dos 3 supervisores definidos ES_1, ES_2 e $SupZ10$:

$$R = SupES_1 \wedge SupES_2 \wedge SupZ10; \tag{68}$$

Conforme visto, os supervisores $SupES_1$, $SupES_2$ e $SupZ10$ são modulares, demonstrado pela condição de não-conflito do comando $NonConflict$, e consequentemente R é $trim$.

De acordo com os teoremas 2 e 3, $L_m(R) = L_m(ES_1) \cap L_m(ES_2) \cap L_m(Z10)$ é controlável em relação a M, então R é um supervisor apropriado para M.

O comportamento em malha fechada da planta M, sob a supervisão de ES_1, ES_2 e $SupCon(M,Z10)$ é idêntico ao obtido com o supervisor monolítico obtido na equação 57:

$$Modular = Supcon(M,R); (90 \text{ estados, } 134 \text{ transições}) \tag{69}$$

Foi apresentado nesse capítulo, os passos de um método para a obtenção de um escalonamento ótimo de tarefas, em uma célula de manufatura, com a utilização de conceitos da TCS e modelos de autômatos temporizados.

Com o uso de ferramentas computacionais, garante-se a precisão dos cálculos, e a obtenção de um controle supervisório de maneira automática.

A utilização de supervisores modulares locais simplifica o processo de atualização, modificação ou manutenção dos supervisores. Cabe observar, porém, que tal abordagem é dependente do teste de não conflito. Eventuais interferências que possam existir entre subsistemas que compartilham eventos, podem causar bloqueios indesejados.

Outras abordagens apresentam eficiência na solução de problemas de escalonamento, porém a sua reutilização nem sempre é possível, devido à rigidez dos algoritmos desenvolvidos. O algoritmo apresentado pode ser generalizado, para a solução de outros problemas, sem a necessidade da intervenção de um especialista.

No próximo capítulo, o método é aplicado para a resolução de um escalonamento em uma célula de manufatura em uma indústria fabricante de elevadores para passageiros, tendo como objetivo a obtenção de uma sequência ótima, considerando o tempo total necessário para a fabricação de um determinado item.

6 ESTUDO DE CASO

Neste capítulo, o método apresentado no capítulo 5, para o escalonamento na produção com a utilização de modelos de autômatos temporizados e conceitos da teoria de controle supervisório modular, é aplicada a uma célula de manufatura em uma indústria de elevadores.

6.1 DESCRIÇÃO DE OPERAÇÃO E MODELAGEM DA PLANTA

A célula de manufatura considerada faz parte de uma linha de produção de portas de pavimento em uma fábrica de elevadores. As portas de pavimento isolam o hall de entrada do elevador e o poço onde o mesmo está instalado. Atuam em conjunto com as portas de cabine, que abrem somente quando a mesma estiver em uma posição segura, nivelada ao pavimento. São compostas basicamente de duas chapas metálicas dobradas, aço carbono pré-pintado ou inox, chamadas de folhas de portas, com reforços fixados na parte interna. Além dos reforços, existem duas peças fixadas nas extremidades superior e inferior, chamadas travessas. As travessas, além de proporcionar resistência estrutural para o conjunto, possuem peças plásticas que atuam como guias, percorrendo trilhos que garantem o alinhamento durante a abertura e fechamento das portas. Essas folhas são montadas sobrepostas, formando o sistema de abertura e fechamento de acesso/saída da cabine.

Para estruturar o conjunto e dar acabamento, as folhas de portas devem ser dobradas, nas suas laterais, no sentido longitudinal.

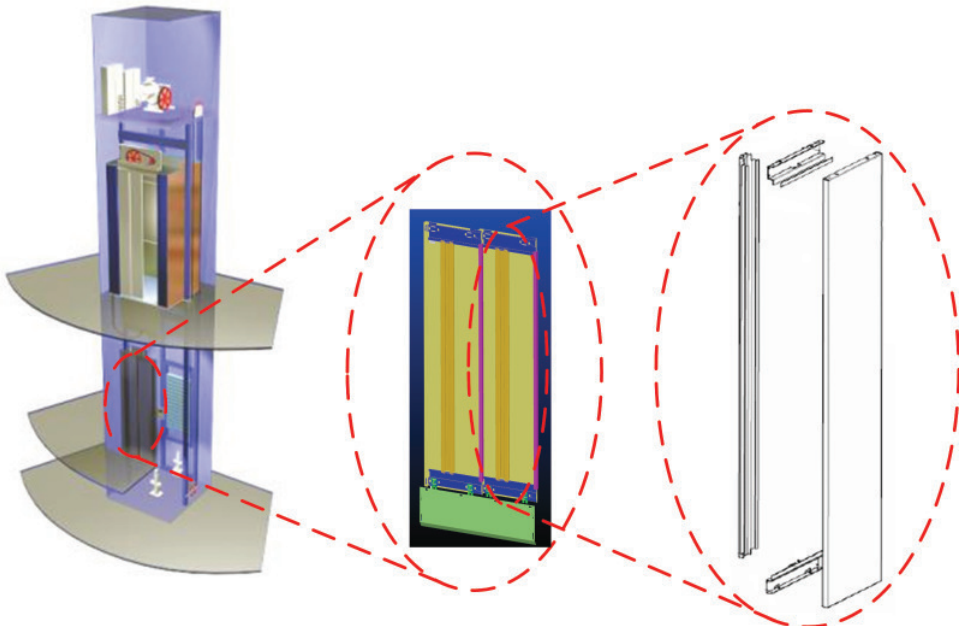

Figura 25 – Elevador de passageiros e detalhes das folhas de porta.

Assim como uma porta residencial, a porta de um elevador possui os chamados batentes, ou colunas, manufaturadas com a mesma matéria prima das folhas de portas. As colunas fazem a fixação do conjunto seja na cabine do elevador, seja na coluna de alvenaria do pavimento.

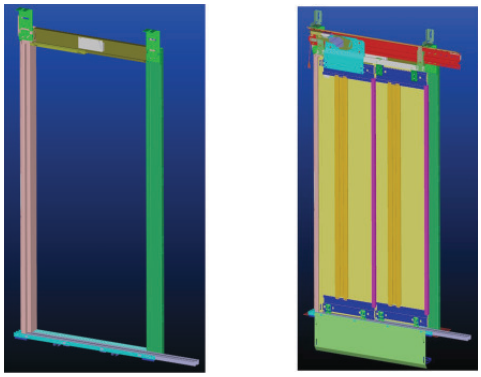

Figura 26 – Detalhe das colunas de portas, e as folhas de portas montadas.

Tanto a folha de porta, como a coluna, são processadas em uma célula composta por cinco equipamentos, conforme mostra a Figura 27: duas esteiras M_0 e M_4, uma dobradeira M_1 e duas máquinas de estampagem M_2 e M_3.

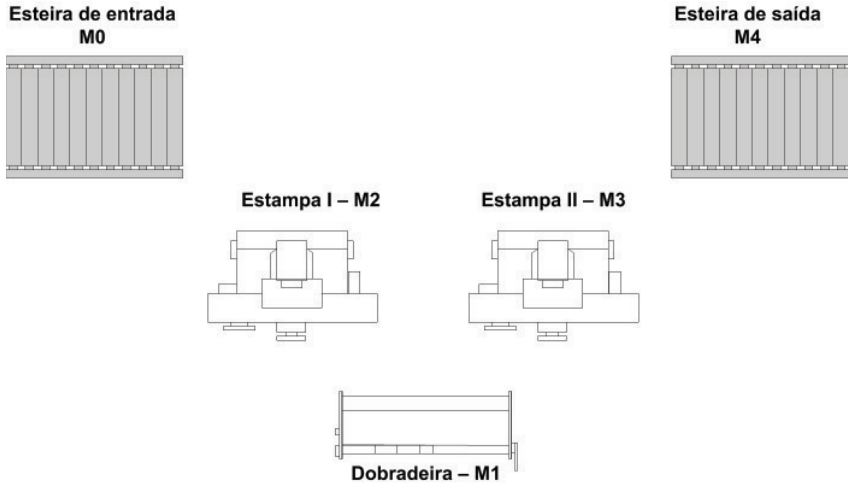

Figura 27 – Célula de estampagem e dobra.

Para formar um sistema de abertura e fechamento de um elevador, são necessários dois conjuntos, cada conjunto composto por uma folha, uma coluna e um reforço. Tanto as folhas de portas, como as colunas, devem passar primeiro pelas máquinas de estampagem, devido à forma construtiva das ferramentas. As colunas, por sua vez, devem ser dobradas primeiro, e somente então os processos de estampagem são feitos.

As setas nas figuras 28, 29 e 30 indicam as etapas do processo pelas quais as chapas metálicas são submetidas:

1. as chapas metálicas chegam à célula por meio da esteira M_0;
2. a colunas são transferidas primeiro para a operação de dobra (M_1), folhas são depositadas primeiro na operação de estampa II (M_3), e reforços na operação de estampa I (M_2);
3. após o término de processamento (uma operação de dobra e duas operações de estampa), as peças são depositadas na esteira M_4;
4. a esteira de saída M_4 retira as peças da célula e encaminha as mesmas para a montagem final da porta.

Neste trabalho, tanto a esteira de entrada quanto a esteira de saída, são consideradas como sendo de capacidade infinita.

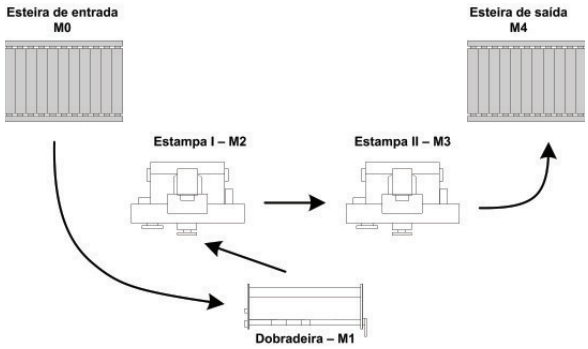

Figura 28 – Sequência de manufatura de uma coluna de porta.

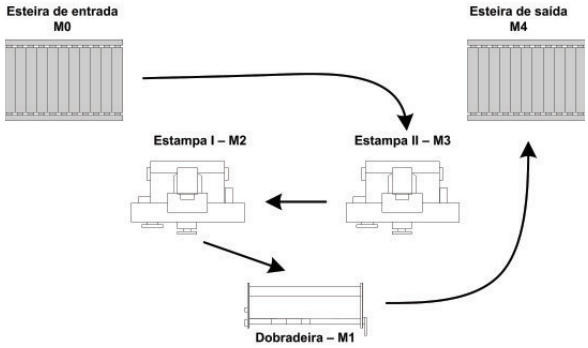

Figura 29 – Sequência de manufatura de uma folha de porta.

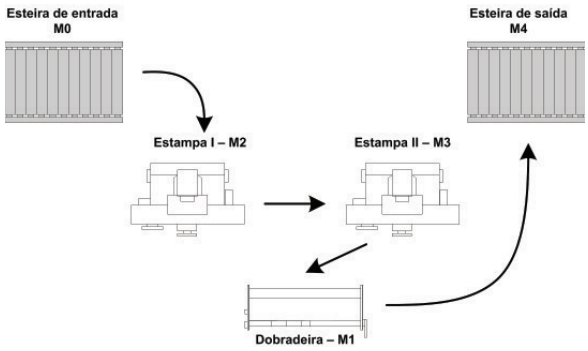

Figura 30 – Sequência de manufatura de um reforço.

A ordem de processamento das peças é apresentada na tabela 2.

Tabela 2 – Sequência de operações.

Peça	Sequência		
coluna (cl)	M_1	M_2	M_3
folha (fl)	M_3	M_2	M_1
reforço (rf)	M_2	M_3	M_1

O motivo pelo qual a estampa na coluna é feita somente após a dobra, deve-se à precisão da máquina, limitada em $\pm$ 0,5 mm em cada dobra executada. Em uma peça, onde são necessárias 4 operações, como a coluna por exemplo, a soma de eventuais imprecisões pode, no pior caso, resultar em uma diferença de até 2 mm com relação à dimensão especificada. Como nas extremidades das colunas, algumas peças devem ser fixadas com a utilização de rebites em orifícios feitos com a estampa, caso haja algum desvio nas medidas, a instalação dos rebites pode não ser possível. Com a estampagem feita na parte final do processo, as dimensões entre os centros dos orifícios, onde serão instalados os rebites, são asseguradas. A figura 31 mostra detalhes das operações de dobra e estampagem de uma coluna.

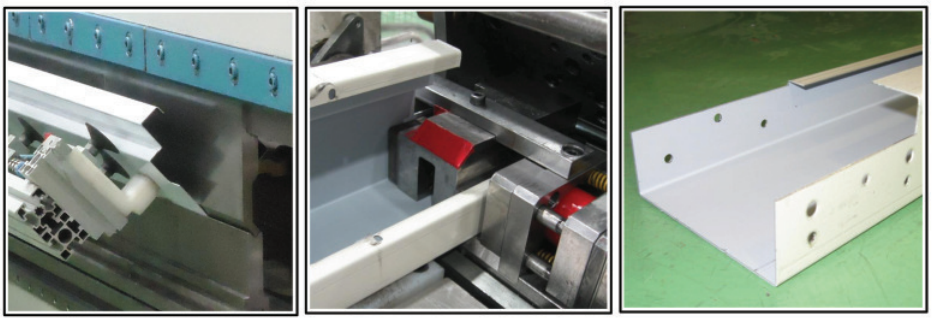

Figura 31 – Operação de dobra e estampa de coluna de porta.

Uma coluna, denominada peça do tipo cl, precisa de 3 unidades de tempo para ser processada na máquina M_1 (dobra), 2 unidades de tempo para ser processada nas máquinas M_2 (estampa I) e M_3 (estampa II). Uma folha de porta, denominada fl, precisa de 1 unidade de tempo para ser processada nas máquinas M_2 e M_3 e 4 unidades de tempo para ser processada na máquina M_1. Um reforço, denominado rf, precisa de duas unidades de tempo para ser processado nas máquinas M_3 e M_1, uma unidade de tempo para ser processado na máquina M_2.

A tabela 3 mostra os tempos das operações de dobra e estampa em cada tipo de peça.

Tabela 3 – Tempos de processamento de dobra e estampa.

	Tempos de Processamento		
	Dobra (M_1)	Estampa I (M_2)	Estampa II (M_3)
cl	3	2	2
fl	4	1	1
rf	2	1	2

Na modelagem em malha aberta de cada um dos subsistemas envolvidos em uma planta, os defeitos que possam surgir nos inúmeros itens que compõem cada equipamento, *i.e.*, motores, sensores, atuadores hidráulicos e/ou pneumáticos, não são os causadores dos problemas operacionais, de coordenação entre os diversos subsistemas (QUEIROZ, 2000). Assim, a modelagem M_i dos componentes da planta considerados na manufatura das peças (M_1, M_2, M_3), feito no nível de abstração que nos interessa, pode ser representado pelos autômatos ATG mostrados na Figura 32.

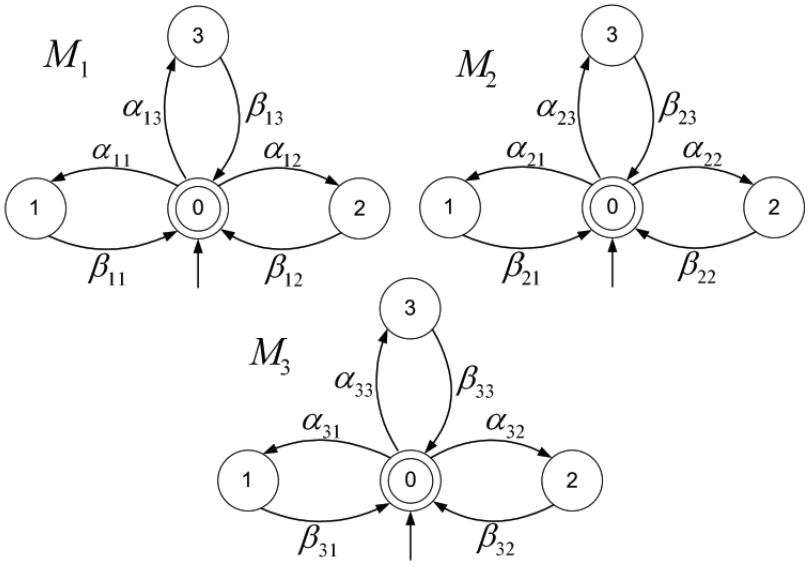

Figura 32 – Autômatos ATG para os subsistemas da planta, M_1, M_2 e M_3.

Os autômatos da figura 32 possuem todos os eventos dos 3 produtos relacionados às máquinas. As máquinas M_1, M_2 e M_3 estão disponíveis somente no estado 0, que é também o estado marcado. Os estados 1, 2 e 3 representam situações em que a máquina está processando os produtos do tipo cl, fl ou rf respectivamente. Após o evento de início de operação, somente após a ocorrência do fim de operação correspondente, é que a máquina volta a ficar disponível novamente.

A nomenclatura dos eventos segue a seguinte estruturação: eventos de início de operação são identificados com a letra α, e os eventos de final de operação com a letra β. Os índices ij de cada evento indicam a máquina i que está processando determinada peça j. Uma coluna, ou cl , é identificada como o número 1. A folha de porta, ou fl é identificada com o número 2, e um reforço, ou rf é identificado com o número 3. Portanto, o evento α_{12} indica que uma peça do tipo 2 (folha de porta, ou fl) está sendo processada na máquina 1 (M_1 - dobra), e a nomenclatura α indica que é o início da operação.

Os eventos de início de operação de cada equipamento fazem parte do conjunto dos eventos forçáveis, $\Sigma_{for} = \{\alpha_{11}, \alpha_{12}, \alpha_{13}, \alpha_{21}, \alpha_{22}, \alpha_{23}, \alpha_{31}, \alpha_{32}, \alpha_{33}\}$, e os eventos de final de operação, fazem parte do conjunto dos eventos não controláveis $\Sigma_{unc} = \{\beta_{11}, \beta_{12}, \beta_{13}, \beta_{21}, \beta_{22}, \beta_{23}, \beta_{31}, \beta_{32}, \beta_{33}\}$.

A tabela 4 sumariza todos os eventos, relevantes para a modelagem do sistema.

Tabela 4 – Eventos de início e final de operação em cada máquina.

		Tipo de Peça		
Máquina	Operação	cl	fl	rf
M_1	Inicia Dobra	α_{11}	α_{12}	α_{13}
M_1	Finaliza Dobra	β_{11}	β_{12}	β_{13}
M_2	Inicia Estampo I	α_{21}	α_{22}	α_{23}
M_2	Finaliza Estampo I	β_{21}	β_{22}	β_{23}
M_3	Inicia Estampo II	α_{31}	α_{32}	α_{33}
M_3	Finaliza Estampo II	β_{31}	β_{32}	β_{33}

Para a modelagem da planta, foi escolhido como o valor do $tick$, um tempo igual a 10 segundos. Na tabela 5, são apresentados os limites de tempo mínimo (l_σ) e máximo (u_σ) de cada operação.

Tabela 5 – Limites de tempos de processamento.

		Limites de tempo	
Máquina	Operação	l_σ	u_σ
M_1	α_{11}	1	∞
	β_{11}	3	3
	α_{12}	1	∞
	β_{12}	4	4
	α_{13}	1	∞
	β_{13}	2	2
M_2	α_{21}	1	∞
	β_{21}	2	2
	α_{22}	1	∞
	β_{22}	1	1
	α_{23}	1	∞
	β_{23}	1	1
M_3	α_{31}	1	∞
	β_{31}	2	2
	α_{32}	1	∞
	β_{32}	1	1
	α_{33}	1	∞
	β_{33}	2	2

O autômato TTG da máquina M_1 obtido partir do autômato *ATG* apresentado na figura 32 é mostrado na figura 33. Esse autômato foi obtido com a função $TimedGraph$ do aplicativo $TTCT$, que considera os intervalos de tempo (l_σ, u_σ) relacionados com cada evento do autômato ATG.

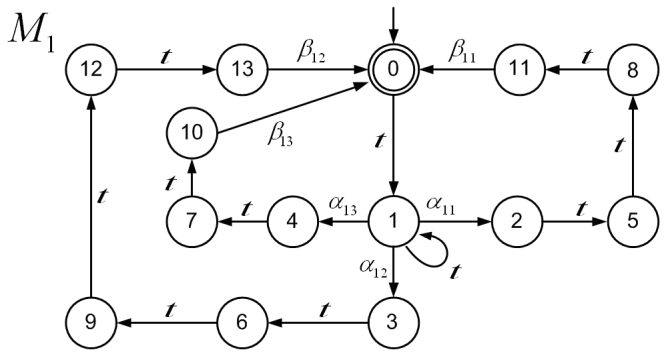

Figura 33 – Autômato TTG para a máquina M_1.

A figura 33 representa o autômato TTG da máquina M_1, com 14 estados e 17 transições, onde observa-se o aumento da complexidade, quando se considera a introdução do evento $tick$ na estrutura, representado pela letra t. Pode-se notar também que, após o início do processamento da peça 1 (evento α_{11}), a finalização da peça (evento β_{11}) ocorre somente após 3 unidades de tempo terem decorrido, conforme a tabela 5, que mostra os valores de u_σ para cada evento.

Os autômatos TTG para as máquinas M_2 e M_3 foram obtidos com o mesmo comando $TimedGraph$, e são mostrados nas figuras 34 e 35. Similarmente ao evento α_{11} e β_{11}, os tempos entre início e fim de operação, para todos os outros eventos, estão vinculados aos valores de u_σ da tabela 5.

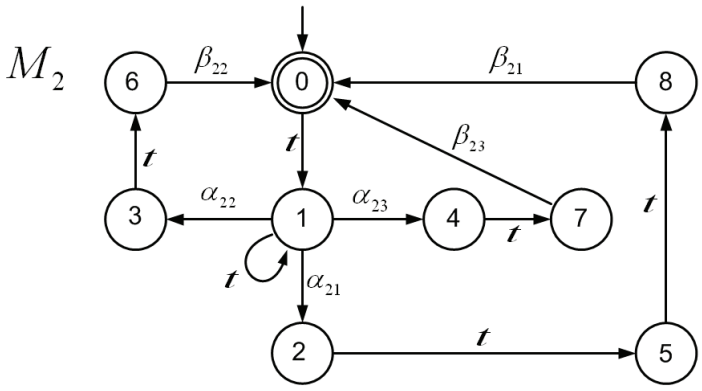

Figura 34 – Autômato TTG para a máquina M_2.

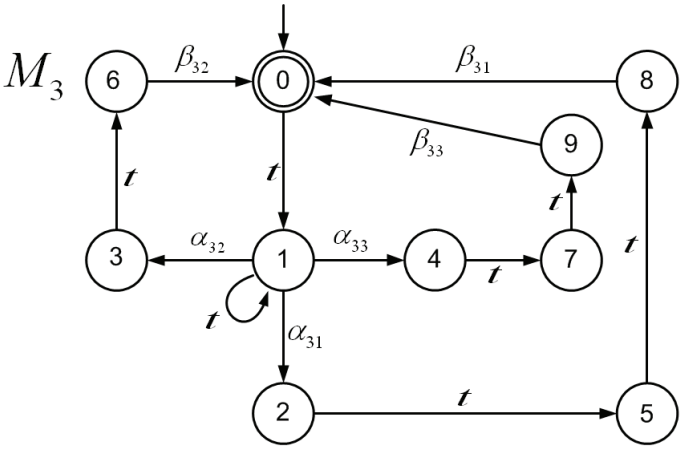

Figura 35 – Autômato TTG para a máquina M_3.

6.2 MODELAGEM DAS ESPECIFICAÇÕES

As restrições, que podem ser modeladas diretamente por meio de um autômato TTG (BRANDIN; WONHAM; BENHABIB, 1992), são dadas pelas seguintes condições, que atendem à sequência de operação especificada na tabela 2:

- uma peça do tipo cl, coluna de porta, deve ser processada primeiro pela máquina M_1, depois pela máquina M_2, e então pela máquina M_3;
- uma peça do tipo fl, folha de porta, deve ser processada pela máquina M_3, M_2 e M_1, nessa ordem, e;
- o reforço, denominado rf, deve ser processado pela máquina M_2, depois M_3 e então M_1.

A primeira restrição pode ser modelada pelo autômato E_1 da figura 36:

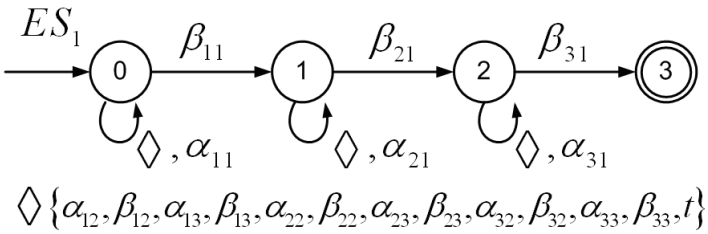

Figura 36 – Autômato para a restrição E_1.

Observa-se da figura 36 que, para que o estado marcado 3 seja atingido, é necessária a ocorrência dos eventos de final de operação β_{11}, β_{21} e β_{31}. Esses eventos representam o final de operação da peça do tipo 1, ou seja, coluna de porta, nas máquinas M_1, M_2 e M_3 respectivamente, conforme a ordem que foi especificada na tabela 2. Assim, o supervisor somente permite sequências que atendam à especificação de ordem de roteiro de manufatura para cada produto. Os eventos de início de operação são modelados, de tal maneira que não ocorram mais de uma vez, antes que a tarefa seja completada.

As outras duas restrições são modeladas de modo semelhante (PINHA; QUEIROZ; CURY, 2010), considerando-se somente a alteração das sequências de fabricação de cada item, e são mostradas nas figuras 37 e 38.

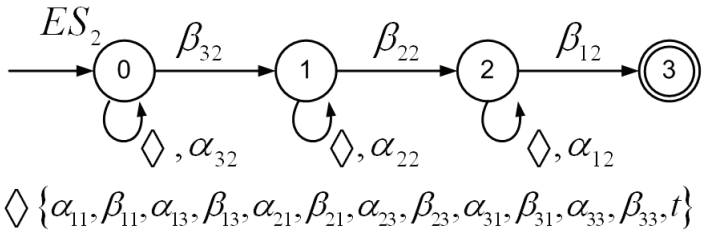

Figura 37 – Autômato para a restrição E_2.

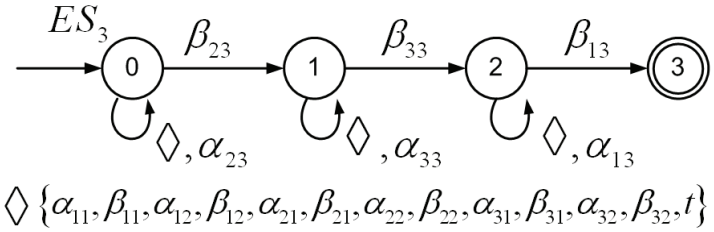

Figura 38 – Autômato para a restrição E_3.

6.3 SÍNTESE DO SUPERVISOR - ABORDAGEM MONOLÍTICA

Conforme visto na seção 5.3, para a obtenção do supervisor conforme a abordagem monolítica, deve-se proceder conforme segue:

i) Obtém-se a planta M resultante da composição paralela dos autômatos ATG de M_1, M_2 e M_3, que descreve o comportamento em malha aberta do sistema. A planta obtida, com o comando $Comp$ do aplicativo TTCT, é um SEDT com um diagrama TTG com 1260 estados e 2598 transições:

$$M = Comp\,(M_1, M_2, M_3)\,; (1260 \text{ estados, } 2598 \text{ transições}) \tag{70}$$

ii) A combinação lógica das especificações, obtida com o comando $Meet$ da ferramenta $TTCT$, e obteve-se uma especificação lógica completa, pela combinação de ES_1, ES_2 e ES_3, formando um SEDT com 64 estados e 352 transições:

$$ES = Meet\,(ES_1, ES_2, ES_3)\,; (64 \text{ estados, } 352 \text{ transições}) \tag{71}$$

iii) Após a obtenção da planta M e da especificação lógica completa ES, obteve-se o comportamento em malha fechada, menos restritivo possível, por meio do comando $Supcon$, um autômato com 2239 estados e 3591 transições:

$$SUP = Supcon(M, ES); (2236 \text{ estados, } 3582 \text{ transições}) \tag{72}$$

6.4 ESCALONAMENTO - ABORDAGEM MONOLÍTICA

Conforme visto na seção 5.3, a especificação de prazo é obtida por meio da composição do supervisor obtido com um temporizador, mostrado na figura 24. Foi imposto à planta um temporizador de tamanho igual a 15 unidades de tempo, obtendo-se uma suprema linguagem controlável $TSUP$, com 4626 estados e 7028 transições. O resultado mostra que existe uma cadeia que atende à restrição de que uma tarefa seja completada em até 15 unidades de tempo.

$$TSUP = Supcon(SUP, Z15); (4626 \text{ estados, } 7028 \text{ transições}) \tag{73}$$

O objetivo, porém, é encontrar uma cadeia que complete uma tarefa no menor tempo possível. Conforme (BRANDIN; WONHAM, 1994), essa condição é obtida ao se efetuar sucessivos cálculos com temporizadores com duração igual a 14, 13,... unidades de tempo, até que o comando $Supcon$ retorne um resultado vazio. O menor ciclo foi obtido com um temporizador com 12 $ticks$:

$$MinSup = Supcon(SUP, Z12); (442 \text{ estados, } 607 \text{ transições}) \tag{74}$$

A tabela 6 mostra os resultados obtidos com os diversos temporizadores:

Tabela 6 – Quantidade de estados e transições.

Supervisor	Estados	Transições
$Supcon(SUP, Z15)$	4626	7028
$Supcon(SUP, Z14)$	2764	4144
$Supcon(SUP, Z13)$	1390	2025
$Supcon(SUP, Z12)$	442	607
$Supcon(SUP, Z11)$	0	0

O resultado da equação 74 comprova que existem sequências factíveis, nas quais o sistema completa uma tarefa em até 12 unidades de tempo, sendo esse o menor ciclo possível.

6.5 SÍNTESE DO SUPERVISOR - ABORDAGEM MODULAR

Conforme visto na seção 5.4, a abordagem modular é obtida com a seguinte sequência de passos:

i) M é a composição paralela de M_1, M_2 e M_3, que foi obtida na equação 70, que resultou em um SEDT com 1260 estados e 2598 transições.

ii) Para que ES_1, ES_2 e ES_3 sejam considerados como supervisores apropriados em relação a M, as condições do Teorema 1, mostrado na seção 5.4 devem ser satisfeitas, i.e., $E_1 = L_m(ES_1)$, $E_2 = L_m(ES_2)$ e $E_3 = L_m(ES_3)$ devem ser controláveis em relação a M. Verificando-se a condição $Trim$ de ES_1, ES_2 e ES_3:

$$SupES_1 = Supcon(M, ES_1); (2238 \text{ estados, } 4544 \text{ transições}) \tag{75}$$

$$SupES_2 = Supcon(M, ES_2); (2566 \text{ estados, } 5039 \text{ transições}) \tag{76}$$

$$SupES_3 = Supcon(M, ES_3); 2688 \text{ estados, } 5267 \text{ transições}) \tag{77}$$

A condição de modularidade de $SupES_1, SupES_2$ e $SupES_3$ é demonstrada pelo comando $NonConflict$:

$$true = Nonconflict(SupES_1, SupES_2) \tag{78}$$

$$true = Nonconflict(SupES_2, SupES_3) \tag{79}$$

$$true = Nonconflict(SupES_1, SupES_3) \tag{80}$$

O comando $Condat$ do aplicativo $TTCT$ é utilizado para verificar se as linguagens representadas por $SupES_1$, $SupES_2$ e $SupES_3$ são controláveis:

$$SupES_1DAT = Condat(M, SupES_1); \text{controlável} \tag{81}$$

$$SupES_2DAT = Condat(M, SupES_2); \text{controlável} \tag{82}$$

$$SupES_3DAT = Condat(M, SupES_3); \text{controlável} \tag{83}$$

Ou seja, como as condições do Teorema 1, seção 5.4 são satisfeitas, então os supervisórios representados por $SupES_1$, $SupES_2$ e $SupES_3$ são apropriados para M.

6.6 ESCALONAMENTO - ABORDAGEM MODULAR

O supervisor relacionado com o tempo mínimo de ciclo, $Z12$, conforme 74, é obtido por meio do comando $Supcon$:

$$SupZ12 = Supcon(M, Z12); (10511 \text{ estados, } 21448 \text{ transições}) \tag{84}$$

E fazendo as mesmas verificações, de ausência de conflito e controlabilidade:

$$true = Nonconflict(SupES_1, SupZ12) \tag{85}$$

$$true = Nonconflict(SupES_2, SupZ12) \tag{86}$$

$$true = Nonconflict(SupES_3, SupZ12) \tag{87}$$

$$SupZ12DAT = Condat(M, SupZ12); \text{controlável} \tag{88}$$

Então SupZ12 é um supervisor apropriado para M.

Obtém-se então o supervisor modular R, resultante da conjunção dos 4 supervisores $SupES_1$, $SupES_2$, $SupES_3$ e $SupZ12$:

$$R = SupES_1 \wedge SupES_2 \wedge SupES_3 \wedge SupZ12; (10351 \text{ estados, } 16271 \text{ transições}) \quad (89)$$

Como a condição de modularidade de $SupES_1$, $SupES_2$, $SupES_3$ e $SupZ12$ foi verificada através do comando $NonConflict$, então R é $trim$.

Além disso, conforme os teoremas 2 e 3, vistos na seção 5.4, $L_m(R) = L_m(ES_1) \cap L_m(ES_2) \cap L_m(ES_3) \cap L_m(SupZ12)$ é controlável em relação a M então R é um supervisor apropriado para M.

A equação 90 representa o comportamento em malha fechada da planta M, sob a supervisão de ES_1, ES_2, ES_3 e $SupCon(M, Z12)$:

$$Modular = Supcon(M, R); (442 \text{ estados, } 607 \text{ transições}) \quad (90)$$

Esse resultado é idêntico ao obtido com o supervisor monolítico na equação 74, comprovando que o supervisor modular baseado nas técnicas apresentadas é equivalente à supervisão centralizada.

6.7 RESULTADOS

A figura 39, mostra uma das sequências de eventos factíveis, que foi obtida por intermédio do método apresentado, i.e., uma das cadeias, que atendem às especificações de ordem de manufatura das peças, e com o menor ciclo possível. Esta cadeia corresponde a uma sequência de operações, que atinge uma tarefa completa após terem decorrido 12 unidades de tempo. Entende-se por tarefa completa a obtenção de um conjunto composto por uma coluna, uma folha de porta e um reforço.

Na figura 39 pode-se observar que:

- Logo após o início, as máquinas M_3, M_1 e M_2 são forçadas a iniciar a operação nas peças do tipo fl, cl e rf, eventos α_{32}, α_{11} e α_{23}, respectivamente. Essas operações ocorrem assim que se tornam elegíveis, e forçadas a ocorrer pelo supervisor, prioritariamente em relação a um outro evento $tick$, por exemplo, que poderia levar o sistema a uma sequência não ótima, i.e, mais longa.

- As máquinas M_3, M_1 e M_2 finalizam as operações das peças fl, cl e rf, eventos β_{32}, β_{11} e β_{23}, após 1, 1 e 3 unidades de tempo terem decorrido, respectivamente.

- Assim que a máquina M_3 finaliza a operação na peça do tipo fl, evento β_{32}, a máquina M_2 é forçada a iniciar a operação na mesma, evento α_{22}.

- Uma máquina inicia uma operação em determinado tipo de peça, somente se a operação iniciada na peça anterior foi finalizada. Por exemplo, a máquina 2 que inicia a operação em uma peça do tipo rf no estado 7, somente irá processar outra peça, após a finalização daquela, o que ocorre somente no estado 32.

- Todos os eventos $\left\{\alpha_{ij}, \beta_{ij}, i, j = 1,2,3\right\}$ ocorrem respeitando os limites de tempo inferior $\{l_\sigma\}$ e superior $\{u_\sigma\}$ mostrados na tabela 5. Ou seja, a finalização da operação de dobra da folha de porta β_{12}, estado 342, ocorre somente decorridos 4 unidades de tempo, após seu início α_{12}, estado 197.

- O escalonamento, ou seja, as sequências de produção das peças, conforme as especificações mostradas na tabela 2, cujos autômatos ES_1, ES_2 e ES_3 foram mostrados nas figuras 36, 37 e 38 , foram respeitadas.

- Uma tarefa completa é executada após 12 unidades de tempo terem decorrido a partir do estado inicial. Quando o sistema atinge o evento marcado 440 da figura, todas as três peças foram produzidas, e esse é o menor ciclo de tempo possível.

Por meio do algoritmo apresentado, foi possível encontrar um escalonamento de tarefas, conforme as restrições impostas de sequência de operações, e também onde o ciclo de tempo necessário para atingir um estado marcado, é o menor possível.

A obtenção de um roteiro de produção, onde o tempo de produção de um determinado lote de peças é mínimo, atende a um dos principais desafios das indústrias de manufatura, que é a constante busca por processos mais eficientes e produtivos.

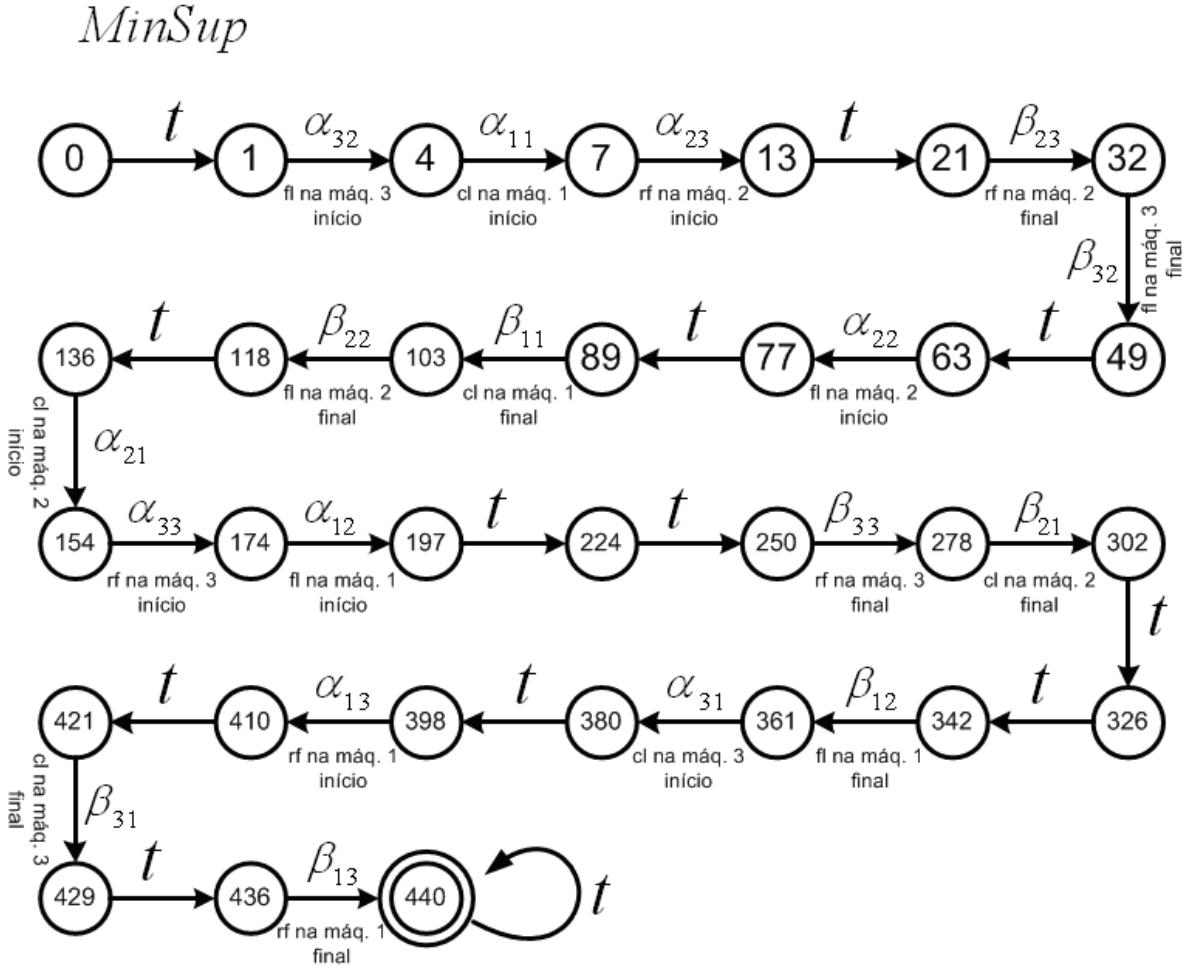

Figura 39 – Mínima sequência de eventos $MinSup$**.**

7 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi apresentada uma proposta para se encontrar um sequenciamento ótimo de operações em uma célula de manufatura, baseado na abordagem de Brandin e Wonham para sistemas a eventos discretos temporizados.

Tal abordagem, a Teoria de Controle Supervisório e suas extensões, conforme apresentada no Capítulo 3, garante a operação ótima de uma planta, considerando situações de bloqueio e eventos proibidos. Porém, a TCS somente, não garante que determinada cadeia de atividades é a melhor possível, tendo-se como critério o tempo total para que uma tarefa completa seja executada.

O método descrito nesta dissertação apresenta a teoria para a obtenção de supervisores para sistemas a eventos discretos temporizados, com a utilização de modelos que incorporam limites de tempo e controles que forçam ou habilitam determinados eventos.

Com a composição de supervisores, ótimos no sentido de serem minimamente restritivos com relação à planta sob controle, e temporizadores que impõem restrições de prazo máximo para que as operações sejam finalizadas, foi possível a obtenção de cadeias que atendem à especificação de prazo.

Assim, a proposta inicial de se estudar problemas relacionados ao escalonamento de tarefas, e utilizar métodos formais da TCS para a obtenção de um sistema de controle, que auxilie na análise de desempenho de células de manufatura, foi satistatoriamente atingida.

Além disso, o procedimento adotado possibilita a prática e a disseminação de tais técnicas no projeto de sistemas automatizados na indústria de manufatura, um ambiente onde nem sempre métodos formais são aplicados.

Como propostas de trabalhos futuros, podem ser citados:

- Extensão do método proposto para problemas de maior complexidade, contando com um maior número de máquinas, ou considerando, por exemplo, quebra e reparação.
- Desenvolvimento de uma ferramenta computacional, para simular e analisar os resultados de escalonamento obtidos, com o objetivo de se analisar em tempo real as sequências de operações que apresentam melhor desempenho.
- Desenvolvimento de aplicativos com uso de linguagens de alto nível $(C^{++}, Java)$, para a conversão dos supervisores e soluções de escalonamento obtidos, para implementação em controladores lógicos programáveis.
- Aplicação da TCS e escalonamento obtido por intermédio de outras abordagens como por exemplo diagramas de decisões binárias (*Binary Decision Diagrams*) (SAADATPOOR; WONHAM, 2007), ou então Redes de Petri Coloridas (EY et al., 2000)
- Desenvolvimento de ferramentas que possibilitem a avaliação de escalonamento, considerando aspectos financeiros, dentro de uma indústria de manufatura. Relacionando-se taxas horárias de produção aos eventos de relógio ($tick$), poder-se-ia determinar a viabilidade econômica de determinados processos, ou então, determinar-se os ganhos financeiros obtidos com escalonamentos mais eficientes e produtivos.

REFERÊNCIAS

ABDEDDAIM, Y.; ASARIN, E.; MALER, O. Scheduling with timed automata. **Theoretical computer science**, v. 354, p. 272–300, 2006.

ASHTON, T. **The industrial revolution, 1760-1830**. Londres: Oxford University Press, 1997.

BACCELLI, F.; G, C.; J, O. G.; QUADRAT, J. Synchronization and linearity, an algebra for discrete event systems. **The journal of the operational research society**, v. 45, n. 1, p. 118, 1992.

BILKAY, O.; ANLAGAN, O.; KILIC, S. Job shop´scheduling using fuzzy logic. **The international journal of advanced manufacturing technology**, v. 23, p. 606–619, 2004.

BRANDIN, B. A.; WONHAM, W. M. The supervisory control of timed discrete event systems. **Proceedings of the 31st IEEE Conference on Decision and Control**, v. 4, p. 3357–3362, 1992.

BRANDIN, B. A.; WONHAM, W. M. Modular supervisory control of timed discrete-event systems. **Proceedings of the 32nd IEEE Conference on Decision and Control**, v. 3, p. 2230–2235, 1993.

BRANDIN, B. A.; WONHAM, W. M. Supervisory control of timed discrete event systems. **IEEE Transactions on automatic control**, v. 39, n. 2, p. 329–342, 1994.

BRANDIN, B. A.; WONHAM, W. M.; BENHABIB, B. Manufacturing cell supervisory control – a modular timed discrete event systems approach. **Proceedings of IEEE International conference on robotics and automation**, v. 2, p. 931–936, 1992.

CASSANDRAS, C. G.; LAFORTUNE, S. **Introduction to discrete event systems. 2nd edition**. New York: Springer, 2008.

CASSANDRAS, C. G.; STRICKLAND, S. G. Sample path properties of timed discrete event systems. **Proceedings of the IEEE**, v. 77, n. 1, p. 59–71, 1989.

CURY, J. E. R. Teoria de controle supervisório de sistemas a eventos discretos. V Simpósio brasileiro de automação inteligente, 2001.

DHINGRA, J. S. **Reconfigurable control in discrete event dynamic systems applied to manufacturing systems**. Tese (Doutorado) — University of Maryland, Maryland, 1992.

EY, H.; SACKMANN, D.; MUTZ, M.; SAUER, J. Adaptive job-shop scheduling with routing and sequencing flexibility using expert knowledge and coloured petri nets. **Proceedings of the 2000 IEEE International conference on systems, man, and cybernetics**, v. 5, p. 3212–3217, 2000.

FENG, L.; WONHAM, W. M. TCT: a computation tool for supervisory control synthesis. **Proceedings of the 8th international workshop on discrete event systems**, p. 388–389, 2006.

FRENCH, S. **Sequencing and scheduling: an introduction to the mathematics of the job-shop**. New York: John Wiley & Sons, 1982.

GOLMAKANI, H. R.; MILLS, J. K.; BENHABIB, B. Deadlock-free optimal routing in flexible manufacturing cells via supervisory control theory. **Proceedings Of The IEEE International Conference On Systems Man And Cybernetics**, v. 4, p. 3390–3395, 2003.

HARRIS, R.; ROTHER, M. Creating continuous flow. Lean enterprise institute, 2003.

HERRMANN, J. W. The legacy of Taylor , Gantt , and Johnson : how to improve production scheduling. **Technical Report**, 2007.

HOLLOWAY, L. E.; KROGH, B. H.; GIUA, A. A survey of Petri nets methods for controlled discrete event systems. **Discrete event dynamic systems**, v. 7, n. 2, p. 151–190, 1997.

JAIN, A. S.; MEERAN, S. A state-of-the-art review of job-shop scheduling techniques. **European journal of operations research**, v. 1, p. 390–434, 1999.

JOHNSON, S. M. Optimal two- and three-stage production schedules with set-up times included. **Naval research logistics quarterly**, v. 1, p. 61–68, 1954.

JONES, A.; RABELO, L. **Survey of job shop scheduling techniques**. Gaithersburg, MD: National institute of standards and technology, 1998.

KLEINROCK, L. **Queueing systems, volume I: Theory**. New York: Wiley-Interscience, 1975.

KROGH, B. H.; HOLLOWAY, L. E. Synthesis for feedback control logic for discrete manufacturing systems. **Automatica**, v. 27, p. 641–651, 1991.

MELO, J. G.; SACOMANO, J. B. Manufatura enxuta, vantagem competitiva baseada na dimensão tempo. **Anais do XII Simpósio de engenharia da produção**, 2005.

MENGCHU, Z. **Modeling and control of discrete-event dynamic systems**. New York: Springer, 2007.

MONTGOMERY, C. A.; PORTER, M. E. **Estratégia – A busca da vantagem competitiva**. Rio de Janeiro: Campus, 1998.

MURATA, T. Petri nets: properties, analysis and applications. **Proceedings of the IEEE**, v. 77, n. 4, p. 541–580, 1989.

NANVALA, H. B. Use of fuzzy logic approaches in scheduling of fms: a review. **International journal on computer science and engineering**, v. 4, p. 1734–1739, 2011.

OGATA, K. **Engenharia de controle moderno**. São Paulo, SP: Prentice-Hall, 2003.

PAULA, M. A. B. d.; SANTOS, E. A. P. Uma abordagem metodológica para o desenvolvimento de sistemas automatizados e integrados de manufatura. **Produção**, v. 18, p. 8 – 25, 2008.

PENA, P. N.; CUNHA, A. E.; CURY, J. E. R.; LAFORTUNE, S. Metodologia e ferramenta de apoio ao teste de não-conflito no controle modular de sistemas a eventos discretos. **Sba: controle & automação**, v. 21, p. 58–68, 2010.

PINHA, D. C.; QUEIROZ, M. H. d.; CURY, J. E. R. Escalonamento da produção com uso da teoria de controle supervisório. **Anais do X Simpósio brasileiro de automação inteligente**, p. 4662–4668, 2010.

PINHA, D. da C. **Escalonamento ótimo baseado na teoria de controle supervisório aplicado a um estaleiro de reparo naval**. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Florianópolis, SC, 2010.

QUEIROZ, M. H. D.; CURY, J. E. R. Modular supervisory control of large scale discrete event systems. In: **Discrete Event Systems: Analysis and Control. Proceedings of the WODES 2000**. [S.l.]: Kluwer Academic, 2000. p. 103–110.

QUEIROZ, M. H. d.; CURY, J. E. R. Controle supervisório modular de sistemas de manufatura. **Sba: controle & automação**, v. 13, n. 2, p. 123–133, 2002.

QUEIROZ, M. H. d.; CURY, J. E. R. Modular supervisory control of large scale discrete-event systems. **Sba: controle & automação**, v. 13, n. 2, p. 123–133, 2002.

QUEIROZ, M. H. de. **Controle supervisório modular de sistemas de grande porte**. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Florianópolis, SC, 2000.

RAMADGE, P. J. G.; WONHAM, W. M. The control of discrete event systems. **Proceedings of the IEEE**, v. 77, n. 1, p. 81–98, 1989.

RODAMMER, F. A.; WHITE, K. P. A recent survey of production scheduling. **IEEE transactions on systems man and cybernetics**, v. 18, n. 6, p. 841–851, 1988.

SAADATPOOR, A. **State Based Control of Timed Discrete Event Systems using Binary Decision Diagrams**. Dissertação (Mestrado) — University of Toronto, Toronto, 2004.

SAADATPOOR, A.; WONHAM, W. M. State based control of timed discrete event systems using binary decision diagrams. **Systems control letters**, v. 56, p. 62–74, 2007.

SACCOL, A. Z.; PEDRON, C. D.; NETO, G. L.; MACADAR, M. A.; CAZELLA, S. C. Avaliação do impacto dos sistemas ERP sobre variáveis estratégicas de grandes empresas no Brasil. **Revista de Administração Contemporânea**, v. 8, p. 9–34, 2004.

SILVA, R. S. Algoritmo clonal para *job shop scheduling* com controle supervisório. **Anais do X Simpósio brasileiro de automação inteligente**, v. 10, p. 1376–1381, 2011.

SLACK, N. **Vantagem competitiva em manufatura: atingindo competitividade nas operações industriais**. São Paulo: Atlas, 1993.

SLACK, N.; CHAMBERS, S.; JOHNSTON, R.; HARLAND, C.; HARRISON, A. **Administração da produção - edição compacta**. São Paulo: Atlas, 2009.

TERRA, A. R. T.; PEREIRA, N. A. Aplicação de redes neurais artificiais na programação da produção. **XX ENEGEP - Encontro nacional de engenharia de produção**, v. 23, p. 606–619, 2000.

WALLACE, C.; JENSEN, P.; SOPARKAR, N. Supervisory control of workflow scheduling. **Proceedings of international workshop on advanced transaction models and architectures**, p. 1–11, 1996.

WANG, X. A language measure for performance evaluation of discrete-event supervisory control systems. **Applied mathematical modelling**, v. 28, n. 9, p. 817–833, 2004.

WILLEMS, T. M.; ROODA, J. E. Neural networks for job-shop scheduling. **Control engineering practice**, v. 2, n. 1, p. 31–39, 1994.

WONHAM, W. M. **Supervisory control of discrete event systems**. University of Toronto, 2011.

WONHAM, W. M.; RAMADGE, P. J. G. Modular supervisory control of discrete event systems. **Mathematics of control of discrete event systems**, v. 1, n. 1, p. 13–30, 1988.

YAHYAOUI, A.; FNAIECH, F. Recent trends in intelligent job shop scheduling. **1st IEEE International conference on e-learning in industrial electronics**, p. 191–195, 2006.

YANG, S.; WANG, D. Constraint satisfaction adaptive neural network and heuristics combined approaches for generalized job-shop scheduling. **IEEE transactions on neural networks**, v. 11, p. 474–485, 2000.

YANG, W. Y.; CHANG, T. G.; SONG, I. H.; CHO, Y. S.; HEO, J.; JEON, W. G.; LEE, J. W.; KIM, J. K. **Signals and Systems with MATLAB**. New York: Springer, 2009.

ZADEH, L. A. Fuzzy sets. **Information and control**, v. 8, n. 8, p. 338–353, 1965.

APÊNDICE A – *SCRIPT* DOS CÁLCULOS FEITOS COM O PROGRAMA $TTCT$, PARA O ESTUDO DE CASO

M1 = ACreate(M1, [mark 0], [time bounds [110,3,3], [111,1,1000], [120,4,4], [12 1,1,1000], [130,2,2], [131,1,1000]], [forcible 111,121,131], [tran [0,111 ,1], [0,121,2], [0,131,3], [1,110,0], [2,120,0], [3,130,0]]) (4,6)

M2 = ACreate(M2, [mark 0], [time bounds [210,2,2], [211,1,1000], [220,1,1], [22 1,1,1000], [230,1,1], [231,1,1000]], [forcible 211,221,231], [tran [0,211 ,1], [0,221,2], [0,231,3], [1,210,0], [2,220,0], [3,230,0]]) (4,6)

M3 = ACreate(M3, [mark 0], [time bounds [310,2,2], [311,1,1000], [320,1,1], [32 1,1,1000], [330,2,2], [331,1,1000]], [forcible 311,321,331], [tran [0,311 ,1], [0,321,2], [0,331,3], [1,310,0], [2,320,0], [3,330,0]]) (4,6)

M = Comp(M1,M2) (16,48)

M = Comp(M,M3) (64,288)

M = Timed Graph(M) (1260,2598)

ALLM = Allevents(M) (1,19)

Create(ES_1, [mark 3], [tran [0,0,0], [0,110,1], [0,111,0], [0,120,0], [0,1 21,0], [0,130,0], [0,131,0], [0,220,0], [0,221,0], [0,230,0], [0,231,0], [0 ,320,0], [0,321,0], [0,330,0], [0,331,0], [1,0,1], [1,120,1], [1,121,1], [1 ,130,1], [1,131,1], [1,210,2], [1,211,1], [1,220,1], [1,221,1], [1,230,1], [1,231,1], [1,320,1], [1,321,1], [1,330,1], [1,331,1], [2,0,2], [2,120,2], [2,121,2], [2,130,2], [2,131,2], [2,220,2], [2,221,2], [2,230,2], [2,231,2], [2,310,3], [2,311,2], [2,320,2], [2,321,2], [2,330,2], [2,331,2], [3,0,3], [3,120,3], [3,121,3], [3,130,3], [3,131,3], [3,220,3], [3,221,3], [3,230 ,3], [3,231,3], [3,320,3], [3,321,3], [3,330,3], [3,331,3]], [forcible 111, 121, 131, 211, 221, 231, 311, 321, 331]) (4,58)

Create(ES_2, [mark 3], [tran [0,0,0], [0,110,0], [0,111,0], [0,130,0], [0,1 31,0], [0,210,0], [0,211,0], [0,230,0], [0,231,0], [0,310,0], [0,311,0], [0 ,320,1], [0,321,0], [0,330,0], [0,331,0], [1,0,1], [1,110,1], [1,111,1], [1 ,130,1], [1,131,1], [1,210,1], [1,211,1], [1,220,2], [1,221,1], [1,230,1], [1,231,1], [1,310,1], [1,311,1], [1,330,1], [1,331,1], [2,0,2], [2,110,2], [2,111,2], [2,120,3], [2,121,2], [2,130,2], [2,131,2], [2,210,2], [2,211,2], [2,230,2], [2,231,2], [2,310,2], [2,311,2], [2,330,2], [2,331,2], [3,0,3], [3,110,3], [3,111,3], [3,130,3], [3,131,3], [3,210,3], [3,211,3], [3,230 ,3], [3,231,3], [3,310,3], [3,311,3], [3,330,3], [3,331,3]], [forcible 111, 121, 131, 211, 221, 231, 311, 321, 331]) (4,58)

Create(ES_3, [mark 3], [tran [0,0,0], [0,110,0], [0,111,0], [0,120,0], [0,1 21,0], [0,210,0], [0,211,0], [0,220,0], [0,221,0], [0,230,1], [0,231,0], [0 ,310,0], [0,311,0], [0,320,0], [0,321,0], [1,0,1], [1,110,1], [1,111,1], [1 ,120,1], [1,121,1], [1,210,1], [1,211,1], [1,220,1], [1,221,1], [1,310,1], [1,311,1], [1,320,1], [1,321,1], [1,330,2], [1,331,1], [2,0,2], [2,110,2], [2,111,2], [2,120,2], [2,121,2], [2,130,3], [2,131,2], [2,210,2], [2,211,2], [2,220,2], [2,221,2], [2,310,2], [2,311,2], [2,320,2], [2,321,2], [3,0,3], [3,110,3], [3,111,3], [3,120,3], [3,121,3], [3,210,3], [3,211,3], [3,220 ,3], [3,221,3], [3,310,3], [3,311,3], [3,320,3], [3,321,3]], [forcible 111, 121, 131, 211, 221, 231, 311, 321, 331]) (4,58)

ES_1 = Sync(ES_1,ALLM) (4,58) Blocked events = None

ES_2 = Sync(ES_2,ALLM) (4,58) Blocked events = None

ES_3 = Sync(ES_3,ALLM) (4,58) Blocked events = None

ES = Meet(ES_1,ES_2) (16,160)

ES = Meet(ES,ES_3) (64,352)

SUP = Supcon(M,ES) (2239,3591)

SUP_Z15 = Supcon(SUP,Z15) (4626,7028)

SUP_Z14 = Supcon(SUP,Z14) (2764,4144)

SUP_Z13 = Supcon(SUP,Z13) (1390,2025)

SUP_Z12 = Supcon(SUP,Z12) (442,607)

SUP_Z11 = Supcon(SUP,Z11) (0,0)

SUP_ES1 = Supcon(M,ES1) (2238,4544)

SUP_ES2 = Supcon(M,ES2) (2566,5039)

SUP_ES3 = Supcon(M,ES3) (2688,5267)

SUP_ES1_DAT = Condat(M,SUP_ES1) Controllable.

SUP_ES2_DAT = Condat(M,SUP_ES2) Controllable.

SUP_ES3_DAT = Condat(M,SUP_ES3) Controllable.

true = Nonconflict(SUP_ES1,SUP_ES2)

true = Nonconflict(SUP_ES2,SUP_ES3)

true = Nonconflict(SUP_ES1,SUP_ES3)

SUPZ12_MODULAR = Supcon(M,Z12) (10511,21448)

true = Nonconflict(SUP_ES1,SUPZ12)

true = Nonconflict(SUP_ES2,SUPZ12)

true = Nonconflict(SUP_ES3,SUPZ12)

SUPZ12_MODULAR_DAT = Condat(M,SUPZ12_MODULAR) Controllable.

R = Meet(ES,SUPZ12_MODULAR) (10351,16271)

MODULAR = Supcon(M,R) (442,607)