UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

# Estudo e Modelagem de Job Shops Cíclicos Com Jobs Distintos

*Autora:* DENISE SODERO VINHAS PORTUGAL

*Orientador:* PROF. DR. RAFAEL SANTOS MENDES

*Co-Orientador:* DR. LUIZ MANOEL AGUILERA

**Banca examinadora:**

Prof. Dr. Rafael Santos Mendes (**Orientador**)
DCA/FEEC/UNICAMP

Prof. Dr. Paulo Morelato França
DENSIS/FEEC/UNICAMP

Prof. Dr. Marcio Rillo
PEE/POLI/USP

Prof. Dr. Luis Gimeno Latre (**Suplente**)
DCA/FEEC/UNICAMP

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, como parte dos pré-requisitos para obtenção do Título de Mestre em Engenharia Elétrica.

**Julho de 1999**

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

P838e

Portugal, Denise Sodero Vinhas
Estudo e modelagem de job shops cíclicos com jobs distintos. / Denise Sodero Vinhas Portugal.--Campinas, SP: [s.n.], 1999.

Orientadores: Rafael Santos Mendes, Luiz Manoel Aguilera.
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Pesquisa operacional. 2. Controle de produção. 3. Modelos matemáticos. I. Mendes, Rafael Santos. II. Aguilera, Luiz Manoel. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

# Estudo e Modelagem de Job Shops Cíclicos com Jobs Distintos

Este exemplar corresponde à redação final da tese devidamente corrigida e defendida por **Denise Sodero Vinhas Portugal** e aprovada pela comissão julgadora.

Campinas,17 de agosto de 1999

---

**Prof. Dr. Rafael Santos Mendes**
**Orientador**

---

**Dr. Luiz Manoel Aguilera**
**Co-orientador**

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação, UNICAMP, como requisito parcial para a obtenção do Título de MESTRE em Engenharia Elétrica.

# Resumo

Um problema de escalonamento cíclico com recursos disjuntivos e múltiplos jobs, denominado job shop cíclico com jobs distintos, é investigado. Para a resolução deste problema duas modelagens em programação inteira mista são adaptadas de modelos encontrados na literatura. Os modelos são implementados no *software* GAMS. O comportamento dos modelos face a múltiplos variações paramétricas é analisado graficamente. Finalmente a equivalência entre os modelos é mostrada analítica e empiricamente.



# Abstract

A cyclic scheduling problem with disjunctive resources and multiples jobs called cyclic job-shop with differents jobs is investigated. The adaptation of two models in mixed integer programming, found in the literature, solve the problem. The models are implemented using software GAMS. The behavior of the models in face of multiple parameter variations is shown graphically. Finally the equivalence between models is shown analytically and empirically.

*A minha querida madrinha Marina*

*in memorian*

# Agradecimentos

A Deus que está sempre presente em minha vida, carregando-me nas horas de maior dificuldade e caminhando ao meu lado em meus momentos felizes;

À minha mãe M. Ester que venceu a sua maior batalha na vida, ultrapassando com sucesso a morte de meu pai Nelson, criando, sozinha, eu e minhas irmãs. Este seu exemplo de esforço e dedicação me inspirou a chegar até aqui;

Ao meu dedicado esposo Rodrigo, por seu amor, amizade e carinho que me fortalecem;

Às minhas irmãs, M. Cecília e Solange, pelas brigas e brincadeiras que só irmãs conhecem e, sobretudo, por me fazerem ultrapassar os meus próprios limites;

Ao professor Rafael, por sua confiança em mim, por sua paciência e dedicação, e por sua habilidade em me orientar. Sua firmeza nas horas certas e seu padrão de qualidade certamente me fizeram crescer;

Ao Luiz Aguilera do Instituto de Automação (IA-CTI), meu co-orientador, pelos ótimos conselhos durante este trabalho, incluindo a minha Iniciação Científica, bem como pelo apoio em geral, que me motivou a seguir esta carreira tão cativante;

Aos amigos de hoje e sempre Akira, Bete, Èrlon, Fernando, Márcia e Marlei, pelo apoio incondicional desde a nossa graduação e à Alessandra que me acompanhou neste mestrado, ajudando-me a ultrapassar barreiras, caminhando sempre em frente rumo ao doutorado;

À Noêmia, por ajudar a diminuir a burocracia que nos assola e por me ajudar a lembrar das datas importantes;

À Unicamp, à FEEC e ao Instituto de Automação (IA-CTI) por sua estrutura e ambiente motivadores;

Ao CNPq, pelo suporte financeiro;

Enfim, a todos aqueles que me ajudaram um pouquinho aqui ou ali

*Agradeço*

# Sumário

# Lista de Tabelas

# Lista de Figuras

# 1 Introdução

Em qualquer tipo de tarefa a repetição pode levar mais perto da perfeição (e também da loucura). Assim quanto mais um jogador de futebol treinar repetidamente cobranças de faltas, maiores serão as possibilidades de se fazer um gol, pois com a repetição vem a observação de detalhes que podem e devem ser melhorados. Esta característica é inerente a qualquer tarefa mecânica e/ou intelectual e pode-se dizer que os aprimoramentos industriais derivam do aprendizado adquirido, através da observação do processo de produção repetitivo.

A interação entre as tarefas cria a necessidade de estabelecer-se uma ordem para a sua execução, dessa forma nenhum jogador coloca a meia por cima da chuteira, mas estabelece uma ordem vestindo primeiro a meia e depois calçando a chuteira. Estas duas tarefas são simples e precisam apenas de bom senso para serem executadas na melhor ordem. Mas existem casos, como em um processo de manufatura, que esta ordem só é conseguida através de estudos específicos para o problema, e para este caso pode-se usar o **Escalonamento** que, de acordo com Baker (1974), é a alocação de recursos no tempo para executar uma coleção de tarefas. Um dos problemas de Escalonamento bem conhecido é o problema de job shop que pode ser definido da seguinte forma: $n$ *jobs* devem ser processados em $m$ diferentes máquinas, cada *job* sendo formado por uma seqüência de operações que devem ser processadas em uma ordem fixa, e cada operação sendo executada em uma única máquina previamente estipulada e com uma duração fixa.

Para se entender melhor este problema em um ambiente de manufatura repetitiva, será feito neste trabalho o estudo e a modelagem de Job Shops Cíclicos com Jobs Distintos. Esta modelagem pode ser utilizada também para resolver alguns problemas na área de computação.

Na literatura é muito encontrado o termo genérico Escalonamento Cíclico, usado para referenciar uma grande quantidade de problemas de escalonamento periódico em um ambiente de demanda contínua e constante, tanto na área de produção quanto na de computação. E o que pode ser observado pelo gradativo aumento de estudos a este respeito é que com o aperfeiçoamento da automação industrial

e da computação paralela, tem sido cada vez maior a busca de melhorias através do Escalonamento Cíclico.

O Escalonamento Cíclico em um ambiente de manufatura não é míope, quando considera um horizonte de tempo infinito para incorporar as necessidades de produção futura e quando examina a interação entre máquinas e operações. Através do comprimento do ciclo, incorpora na solução o que acontece em estado regular[1], podendo fornecer informações valiosas a respeito do fluxo do sistema e dos tempos de fluxo dos *jobs*. Quando aplicável, é possível ter-se um controle mais fácil dos operários, graças a estrutura e a repetição impostas na instalação destes escalonamentos. Isto permite aos programadores de produção concentrar seus esforços em reduzir a ineficiência da manufatura envolvendo tempos de configuração de máquinas improdutivas ou problemas de qualidade/rendimento.

Em um ambiente computacional, uma aplicação importante é a execução de laços (*loops*) de vetores por computadores paralelos onde o corpo do laço define o conjunto de tarefas genéricas. A solução do problema de escalonamento cíclico provê um programa eficiente a ser executado por computadores em paralelo. Além disso, algoritmos de resolução podem ser implementados em compiladores de linguagem de alto nível. Um exemplo disso é a aplicação do mapeamento de um laço de vetores recorrentes numa arquitetura de Termo de Instrução Muito Grande (*Very Large Instruction Word*).

O problema de job shop cíclico com *jobs* distintos pode ser considerado como sendo um caso particular do problema de escalonamento cíclico. Para solucioná-lo duas formulações em programação inteira mista, encontradas na literatura (Rao, 1992 e Hanen, 1994), serão adaptadas neste trabalho, algumas características do problema serão estabelecidas e estudadas como a altura de recorrência entre operações e a largura e o comprimento dos *jobs*. Para implementar as formulações estudadas neste trabalho será utilizado o software GAMS, que é uma linguagem de alto nível para a formulação de modelos de Pesquisa Operacional. A equivalência entre as novas formulações também será examinada.

Este trabalho será então estruturado da seguinte forma: as definições básicas são dadas no Cap. 2 que apresenta o problema de escalonamento de *jobs* idênticos e as formulações encontradas na literatura a serem utilizadas. No Cap. 3 são feitas a descrição e as adaptações das formulações para o problema de *jobs* distintos. No Cap. 4 mostram-se os resultados experimentais que indicam a equivalência dos modelos adaptados. No Cap. 5 é feito um estudo de caso, apresentando-se dois exemplos cujos resultados são comparados utilizando-se os métodos propostos no Cap. 3. Finalmente no Cap. 6 é feita a prova analítica da equivalência entre as duas novas formulações.

[1]Estado em que a execução de todas as operações do Sistema é periódica, com o mesmo tempo de período.

# 2

# Problema de Escalonamento Cíclico

O estudo do problema de escalonamento cíclico (PEC) tem sido realizado sobre diversos pontos de vista, dependendo da aplicação desejada, tanto na área de produção quanto na de computação. Alguns autores que são referências básicas para estudos nessa área, definiram ferramentas teóricas relevantes para solucionar alguns tipos de PEC. Este é o caso do estudo de Dauscha *et al.* (1985) sobre o problema de sequenciamento cíclico com uma aplicação para o controle de tráfego e que será melhor detalhado mais adiante neste capítulo; também é o caso do estudo feito por Serafini & Ukovich (1989) relativo a problemas de escalonamentos cíclicos sendo uma das referências do artigo de Hanen (1994) que será discutido neste trabalho; e ainda é o caso do estudo feito por Roundy (1992) que estabeleceu uma caracterização da "estrutura de precedência cíclica" que é usada para desenvolver um procedimento de busca **Branch & Bound** especializado para identificar este tipo de estrutura. Este estudo foi referência de Rao (1992) e será discutido neste capítulo. O problema de Job Shop Cíclico é NP-difícil. Diversas anaálises de complexidade podem ser encontradas na literatura (Kamoun & Sriskandarajah, 1993; Munier, 1996 e Verhaegh *et al.*, 1998).

Uma das aplicações mais conhecidas, mais antigas e mais simples, pois não depende tanto de tecnologias avançadas, é o provimento de pessoal através do escalonamento de operários em fábricas, de enfermeiras em hospitais, da tripulação em aviões, ou ainda, de operadores de mesa telefônica (Bartholdi III et al., 1980; Bartholdi, 1981 e Emmons, 1985). Problemas clássicos da área de escalonamento, como o job shop e o flow shop, foram estudados em um ambiente de demanda contínua e constante, usando-se ferramentas do PEC. Assim tem-se o problema de flow shop estudado por Karabati *et al.* (1992) que fizeram uma abordagem especial em problemas de escalonamento cíclico com *buffers*[1] de capacidade finita, e também o de Kats & Levner (1997) que estudaram um problema de escalonamento cíclico de *jobs* idênticos em um flow shop reentrante. O problema de *job shop* tem sido muito pesquisado. Assim Yura (1989) mostrou a necessidade de uma capacidade de *buffer* para haver um escalonamento

[1] Local de armazenagem provisória

sem ociosidade; Roundy (1992) estudou a estrutura combinatorial dos escalonamentos cíclicos e provou que sua complexidade é NP-difícil; Hanen (1994) fez uma abordagem de um problema de escalonamento cíclico com recursos disjuntivos, chamado job shop recorrente, e que será discutido ao longo deste trabalho; para problemas de escalonamento de job shops periódicos, Song & Lee (1996) desenvolveram um procedimento de busca tabu; enquanto Lee & Posner (1997) escalonaram e fizeram medidas de performance de alguns desse problemas; e também Song & Lee (1998) modelaram uma rede de Petri para solucionar um job shop periódico com bloqueio e propuseram também uma modelagem em programação inteira mista para achar um escalonamento ótimo livre de impasse (deadlock) e com ciclo mínimo. Foram analisados também os problemas de *lot-sizing* e do caixeiro viajante apresentados em problemas de lote cíclico com tempos de configuração de máquina dependente da seqüência (Dobson, 1992 e Carter *et al.*, 1988). Orlin (1982) analisou um problema de roteamento de veículo e solucionou, em tempo polinomial, o problema de minimizar o número de veículos para satisfazer um escalonamento periódico fixo. E ainda Park & Yun (1985) desenvolveram um modelo em programação inteira mista para escalonamentos de atividades múltiplas, que requerem processamento periódico em um único processador.

Com o avanço da tecnologia, é cada vez maior a busca de melhorias através do uso do escalonamento cíclico. A diversidade deste uso vai desde o planejamento de produção numa política de *just-in-time*, onde a observação de tarefas repetitivas é muito importante para o entendimento do processo de produção e conseqüente aperfeiçoamento (Hall, 1988; Nori & Sarker, 1996 e Tayur, 1996), passando pela minimização de custos de operação e serviços de manutenção (Bar-Noy *et al.*, 1998) e pela transmissão de dados de rádio periódicos (Gondhalekar, 1995), chegando ao transporte de partes de produtos entre máquinas, pois, com uma produção cíclica, o transporte das peças entre as esteiras também deve ter um escalonamento cíclico ótimo, e este transporte pode ser feito através de robôs (Crama & Klundert, 1997 e Kats & Levner, 1997) ou por outros meios (Lei & Wang, 1991; Lei *et al.*, 1993).

Audsley & Burns (1990) mostraram resultados recentes da aplicação da teoria de escalonamento em sistemas em tempo real, incluindo resultados em processos periódicos. Foram propostos modelos em programação não linear inteira mista para encontrar um escalonamento cíclico tanto em linhas de produção paralelas e contínuas em uma indústria química (Sahinidis & Grossmann, 1991), quanto na produção em várias etapas de multiprodutos contínuos (Pinto & Grossmann, 1994). Zhang & Graves (1997) estudaram o comportamento de um escalonamento cíclico em um ambiente estocástico caracterizado por quebras aleatórias de máquinas, que poderiam atrasar a execução de tarefas e como

conseqüência atrapalhar o escalonamento cíclico especificado.

Vários estudos teóricos, a respeito do problema de escalonamento cíclico em um ambiente de manufatura de produção periódica, teêm sido realizados, como em Grigor'Yeva *et al.* (1989) que fizeram um levantamento a respeito dos modelos de escalonamento que podem ser utilizados; Loerch & Muckstadt (1994) analisaram o planejamento de produção e o escalonamento em sistemas de manufatura escalonados ciclicamente, enquanto Maxwell *et al.* (1986) estudaram a interação entre o planejamento e o escalonamento em ambiente de manufaturas repetitivas. Modelos, formulações e heurísticas para o PEC em linhas de fluxo foram discutidos por Karabati & Kouvelis (1996), Rao (1992) e Rao (1993). O escalonamento em eventos periódicos foi estudado por Burkard (1986) e também por Vince (1989) que fez uma pequena análise a respeito de um escalonamento mais eficiente de trens. Chrétienne (1991) mostrou que a noção de escalonamento atrasado (*latest*), que é bem conhecida e útil na teoria de escalonamento clássico, pode ser estendida para um PEC com datas de entrega de produtos (*deadlines*).

O problema de produção cíclica foi estudado como uma aplicação da max-algebra (Changyou, 1990; Lee, 1994; Groth & Santos-Mendes, 1997 e Nawijn, 1998); e também através de teoria dos grafos (Hanen, 1994; Munier, 1996b e Levner & Kats, 1998).

Com os avanços da computação paralela e das arquiteturas dos computadores, novas aplicações para o escalonamento cíclico estão sendo encontradas, este é o caso do estudo feito por Risset *et al.* (1997) sobre o escalonamento de equações de recorrência de sistemas estruturados; também é o caso de Calland *et al.* (1995) que apresentam uma heurística para a resolução de um problema de *software pipelining*; e de Gaujal (1994) que pesquisou um problema de seqüências regulares; e ainda de Hanen (1989) que utilizou redes de Petri para otimizar um problema de arquitetura canalizada (*pipelined*), e em seus estudos a respeito de seqüências regulares sugeriu aplicações em escalonamentos cíclicos. Munshi & Simons (1990), Hanen & Munier (1992) e Hanen & Munier (1993) estudaram as propriedades e características de máquinas paralelos ou recursos múltiplos na paralelização de laços de vetores. Gasperoni & Schwiegelshohn (1991) propuseram algoritmos para solucionar problemas de escalonamento sem interrupção de um conjunto cíclico de operações interdependentes, representado por um exemplo de um laço de programa, com $p$ processadores idênticos.

Uma boa descrição do PEC é dada por Dauscha *et al.* (1985): "Um dado conjunto finito de tarefas deve ser escalonado em um fluxo contínuo e periódico, isto é, todas as tarefas devem ser repetidamente executadas de forma que o intervalo de tempo entre dois pontos de ativação sucessiva de uma mesma tarefa é uma constante $Z$ para todas as tarefas. Esta constante $Z$ será tratada por comprimento do período (ou do ciclo) e uma atribuição dos tempos de ativação das tarefas por escalonamento cíclico."

O problema de job-shop recorrente (Hanen, 1994) e o problema de escalonamento cíclico de *jobs* idênticos (Rao, 1992) combinam as restrições do problema de escalonamento básico e máquinas especializadas[2] (*special-purpose*). O objetivo é achar um escalonamento periódico com um fluxo máximo, o que equivale a achar um escalonamento periódico com um período mínimo. Tanto em Rao (1992) quanto em Hanen (1994) os tempos de processamento das operações são fixos, não existindo sobreposição de operações em uma mesma máquina e também cada operação é associada a uma única máquina. Estas características correspondem as do problema estudado neste trabalho, e por isso estes modelos serão mais detalhados nas duas próximas seções. Apesar dessa similaridade, os modelos de Hanen (1994) e Rao (1992) não são equivalentes pois cada um otimiza o fluxo de *jobs* de forma diferente, sem uma correspondência direta, como será mostrado nas próximas seções. E como essas modelagens não tratam do caso de *jobs* distintos estudado neste trabalho, no próximo capítulo serão feitas alterações nesses modelos para que se adeqüem ao problema de escalonamento cíclico de *jobs* distintos, e será mostrado também que estas duas novas formulações serão equivalentes.

## 2.1 Modelagem do Escalonamento Cíclico de *Jobs* Idênticos (ECJI)

Rao (1992) em seu estudo sobre escalonamento cíclico, propôs uma formulação em programação inteira mista para o problema de *jobs* idênticos visando resolver um problema em um ambiente de manufatura, onde processa-se repetidamente um Conjunto de Peças Minimal ($MPS$) de *jobs*. Cada *job* no $MPS$ consiste de um conjunto de operações a serem executadas em alguma ordem. Foi assumido que os atributos das operações incluem: uma única máquina habilitada $m_i$, um tempo de processamento fixo $p_i$ e um único sucessor no roteamento do *job* $s_i$. Estudou também a relação entre o escalonamento cíclico e dois subproblemas, o de momento de produção, (*production timing*), e o de contrabalanceamento de ciclo (*cycle offset*).

Um escalonamento cíclico será completamente especificado pelo comprimento do ciclo $Z$ e uma coleção de tempos de início das operações $T_i$, medidos em módulo $Z$, tal que:

- não possam existir sobreposições de operações;
- uma vez que a operação comece a ser processada, não possa ser interrompida para execução de outra operação;
- todas as operações serão realizadas exatamente um única vez em qualquer intervalo de comprimento $Z$.

[2]Uma operação qualquer só pode ser processada na máquina associada a ela

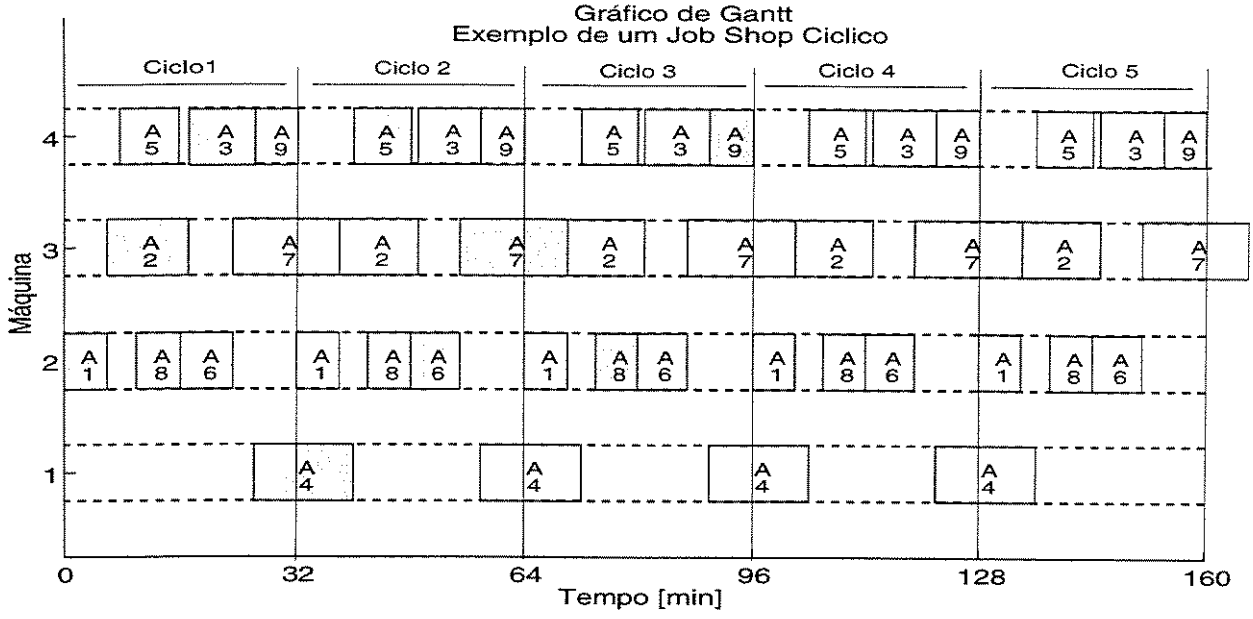



Figura 2.1: Exemplo de um Job-Shop Cíclico

Note-se que cada execução de uma operação $i$ pode ser associada a algum número de ciclo. Assim como pode ser visto na figura 2.1, a primeira execução da operação 1 aparece no ciclo 1, a segunda aparece no ciclo 2, assim por diante. A variável de contrabalanceamento de ciclo $O_i$ mede a diferença entre o número de ciclo da operação $i$ e o número de ciclo de sua operação sucessora $s_i$, em um *job* específico. Assim o contrabalanço de ciclo varia entre os números 0, 1 e 2, logo $O_i \leq 2$.

As notações e as variáveis de decisão utilizadas na formulação de Rao (1992) são descritas a seguir:

- $N$ : conjunto das operações do *job*;
- $i, j, n$ : índices das operações associadas ao *job*;
- $\mathbb{M}$ : conjunto de máquinas;
- $m, s$ : índices das operações associadas às máquinas;
- $m_i$ : única máquina capaz de processar a operação $i$;
- $s_i$ : único sucessor da operação $i$;
- $\Theta_m$ : conjunto de operações associadas a máquina $m$;
- $\sigma_i$ : sucessor imediato da operação $i$ na máquina $m_i$;

- $p_i$ : tempo de processamento da operação $i$;
- $Z$ : comprimento do ciclo;
- $X_{ij}$ : supondo $m_i = m_j$ esta variável está associada ao sequenciamento das tarefas $i$ e $j$ na máquina correspondente sendo definida por:

$$X_{i,j} = \begin{cases} 1 & \text{se } m_i = m_j \text{ e } i \text{ precede imediatamente } j \text{ na máquina } (j = \sigma_i); \\ 0 & \text{caso contrário } (j \neq \sigma_i). \end{cases}$$

- $O_i$ : equilíbrio do ciclo, *cycle offset*, entre as operações $i$ e $s_i$ do mesmo $job$.

$$O_i = \begin{cases} 0 & \text{se } s_i \text{ é inicializado no mesmo ciclo que a operação } i, \\ 1 & \text{se } s_i \text{ é inicializado no ciclo seguinte relativo ao início da operação } i, \\ 2 & \text{caso contrário.} \end{cases}$$

- $L_i$ : um indicador para a operação considerada a última do ciclo na máquina.

$$L_i = \begin{cases} 1 & \text{se } i \text{ é a última operação da máquina } m_i \text{ no ciclo;} \\ 0 & \text{caso contrário.} \end{cases}$$

- $t_i$ : tempo de início de execução da operação $i$;
- $T_i$ : $t_i \mod Z$ [3];

O problema tem como objetivo minimizar o período $Z$ e seu conjunto de restrições é definido da seguinte maneira:

- Para satisfazer a ordem de precedência especificada, utiliza-se:

$$T_{s_i} - T_i \geq p_i - O_i Z \quad i = 1, \ldots, N-1. \tag{2.1}$$

- Para garantir a não sobreposição de tarefas em um máquina qualquer, utiliza-se:

$$T_{\sigma_i} - T_i \geq p_i - L_i Z \quad i = 1, \ldots, N, \tag{2.2}$$

$$\sum_{i \in \Theta_m} L_i = 1 \quad \forall m \in \mathbb{M}. \tag{2.3}$$

[3] $a \mod b$ = resto da diviso de a por b

- Para que não haja subseqüências cíclicas utiliza-se a restrição a seguir e que é melhor explicada no Anexo A:

$$\sum_{(i,j)\in S} X_{ij} \leq |S| - 1 \quad \forall S, \quad S \subset \Theta_m, \mathbb{O}_{s'} \neq \Theta_m, \quad \forall m \in \mathbb{M}. \tag{2.4}$$

- O seguinte conjunto de equações será utilizado para garantir a unicidade de predecessor e sucessor das operações associadas à mesma máquina:

$$\sum_{i:m_i=m_j} X_{ij} = 1 \quad \forall j, \tag{2.5}$$

$$\sum_{j:m_j=m_i} X_{ij} = 1 \quad \forall i. \tag{2.6}$$

- Determina-se que o *job* seja inicializado no tempo 0, que os tempos de inicialização de todas as tarefas sejam sempre positivos e medidos com relação ao período $Z$, isto é, módulo $Z$, e por último que o ciclo seja estritamente positivo:

$$T_1 = 0. \tag{2.7}$$

$$T_i \geq 0, \quad \forall i \tag{2.8}$$

$$T_i < Z. \quad \forall i \tag{2.9}$$

$$Z > 0. \tag{2.10}$$

O conjunto de variáveis combinatóriais ($X_{ij}$, $O_i$ e $L_i$) caracterizando um escalonamento cíclico é denominada estrutura de precedência cíclica (Roundy, 1992).

Na restrição 2.2, $\sigma_i$ é desconhecido e assim pode-se reescrever essa equação em termos das variáveis $X_{ij}$ como segue:

$$T_k - T_i \geq p_i - L_i Z - (1 - X_{ik})B, \quad \forall i, \forall k \neq i, m_k = m_i. \tag{2.11}$$

Deve ser observado que desde que $Z$, $O_i$ e $L_i$ são variáveis, as restrições 2.1 e 2.11 são não lineares. Estas equações podem ser facilmente linearizadas, usando-se técnicas de programação inteira, sem que haja a perda do valor ótimo do problema. Pode-se associar a variável $O_i = \{0, 1, 2\}$ a uma soma de duas variáveis binárias, isto é:

$$O_i = O_i^x + O_i^y \quad i = 1, \ldots, N.$$

E o seguinte conjunto de restrições será proposto:

$$T_{s_i} - T_i \geq p_i - (O_i^x + O_i^y)B \quad i = 1, \ldots, N-1; \tag{2.12}$$

$$T_{s_i} - T_i \geq p_i - Z - O_i^x B \quad i = 1, \ldots, N-1; \tag{2.13}$$

$$T_{s_i} - T_i \geq p_i - Z - O_i^y B \quad i = 1, \ldots, N-1; \tag{2.14}$$

$$T_{s_i} - T_i \geq p_i - 2Z \quad i = 1, \ldots, N-1; \tag{2.15}$$

$$O_i = O_i^x + O_i^y \quad i = 1, \ldots, N. \tag{2.16}$$

Este conjunto corresponde a todas as combinações possíveis entre duas operações na mesma máquina, assim se $O_i = 0$ então $O_i^x = O_i^y = 0$ e a restrição 2.12 corresponderá a equação 2.1 e as outras restrições neste caso estarão satisfeitas; se $O_i = 1$ então ou $O_i^x = 1$ ou $O_i^y = 1$, assim ou a restrição 2.13 ou a restrição 2.14 corresponderá a equação 2.1 isto faz com que as outras restrições estejam satisfeitas; e se $O_i = 2$ então $O_i^x = O_i^y = 1$, e a restrição 2.15 será equivalente a equação 2.1 e mais uma vez as outras restrições estarão satisfeitas. Portanto a equação 2.1 pode ser substituída por este conjunto de restrições.

Da mesma forma a restrição 2.11 pode ser substituída por:

$$T_k - T_i \geq p_i - Z - (1 - X_{ik})B, \quad \forall i, \forall k \neq i, m_k = m_i; \tag{2.17}$$

$$T_k - T_i \geq p_i - L_i B - (1 - X_{ik})B, \quad \forall i, \forall k \neq i, m_k = m_i. \tag{2.18}$$

E a formulação do ECJI poderá ser reescrita de seguinte forma:

$$\begin{aligned}
&\text{Min } Z \\
&T_{s_i} - T_i \geq p_i - (O_i^x + O_i^y)B \quad i = 1, \ldots, N-1; \\
&T_{s_i} - T_i \geq p_i - Z - O_i^x B \quad i = 1, \ldots, N-1; \\
&T_{s_i} - T_i \geq p_i - Z - O_i^y B \quad i = 1, \ldots, N-1; \\
&T_{s_i} - T_i \geq p_i - 2Z \quad i = 1, \ldots, N-1; \\
&O_i = O_i^x + O_i^y \quad i = 1, \ldots, N; \\
&T_k - T_i \geq p_i - Z - (1 - X_{ik})B, \quad \forall i, \forall k \neq i, m_k = m_i; \\
&T_k - T_i \geq p_i - L_i B - (1 - X_{ik})B, \quad \forall i, \forall k \neq i, m_k = m_i. \\
&\sum_{i \in \Theta_m} L_i = 1 \quad \forall m; \\
&\sum_{(i,j) \in S} X_{ij} \leq |S| - 1 \quad \forall S, \quad S \subset \Theta_m, \mathbb{O}_{s'} \neq \Theta_m, \quad \forall m; \\
&\sum_{i : m_i = m_j} X_{ij} = 1 \quad \forall j; \\
&\sum_{j : m_j = m_i} X_{ij} = 1 \quad \forall i;
\end{aligned} \tag{2.19}$$

$$
\begin{aligned}
T_1 &= 0; \\
T_i &\geq 0, \quad \forall i \\
T_i &< Z. \quad \forall i \\
Z &> 0.
\end{aligned}
$$

Esta formulação em programação inteira mista (MILP) foi introduzida no artigo de Rao (1992) somente para formalizar a relação entre certos subproblemas e o escalonamento cíclico. Desde que qualquer problema combinatório é computacionalmente difícil, quando trabalha-se com problemas de grande porte, técnicas simples de programação inteira são impraticáveis nestas circunstâncias. Ainda assim, esta programação matemática pode ser facilmente estendida para incorporar outras questões de interesse, como, por exemplo, o problema de múltiplos *jobs* distintos que será explorado no próximo capítulo.

Para visualizar melhor o problema discutido nesta seção, será apresentado um pequeno exemplo, que será modelado usando-se a formulação proposta por Rao (1992), sua resolução será feita utilizando-se o software GAMS, que é melhor explicado no Apêndice C.

**Exemplo 2.1** : *A tabela a seguir mostra os tempos de processamento das operações de um job A obedecendo uma rota pré determinada* [4]*:*

| Job | Operação | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| A | (11,2) | (13,4) | (6,3) | (11,4) | (1,1) | (11,5) | (8,3) |

Tabela 2.1: Dados do exemplo 2.1

O objetivo é minimizar o período para a repetição das operações.

O comprimento de ciclo ótimo encontrado foi de 24 unidades de tempo (u.t.) e o escalonamento cíclico ótimo é dado pela figura 2.2. Observe-se que a máquina 4 está sendo totalmente utilizada, por isso o comprimento do ciclo não poderá ser menor que 24 u.t.. E ainda, os três primeiros períodos podem ser associados a um estado transitório que levará ao estado regular a partir do quarto período, quando então todas as operações do *job* irão aparecer periodicamente no escalonamento.

[4] Onde $(p_i, m_i)$ = (tempo de processamento da operação $i$, máquina associada a operação $i$)

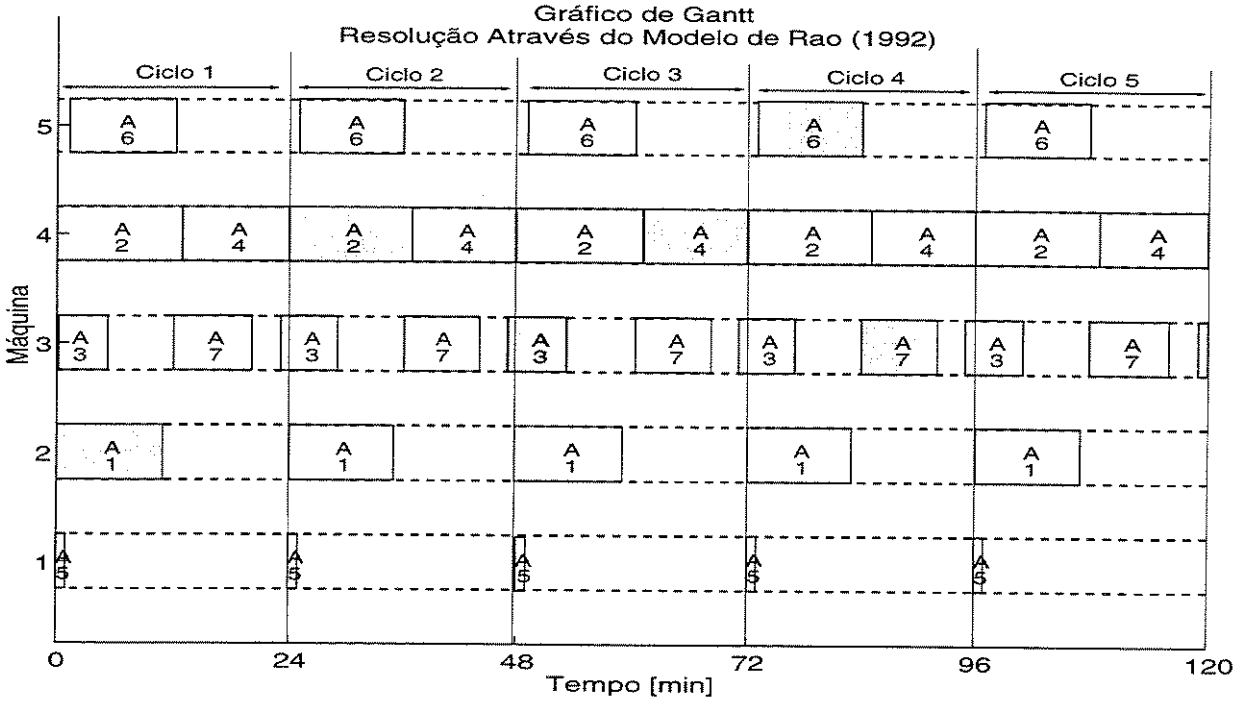



Figura 2.2: Solução ótima de um Job-Shop Cíclico de Jobs Idênticos

## 2.2 Modelagem do Job-Shop Recorrente (JSR)

Hanen (1994) em seu estudo sobre escalonamento cíclico, propôs uma formulação em programação inteira mista para o problema de escalonamento cíclico com recursos disjuntivos[5] e que denominou job-shop recorrente (JSR).

Como qualquer problema de escalonamento cíclico, o job-shop recorrente é caracterizado por um conjunto de tarefas genéricas, que são executadas infinitas vezes. Estas tarefas são não-reentrantes, isto é duas ocorrências de uma tarefa genérica não se sobrepõem.

O conjunto de tarefas genéricas é denominado por $T = 1, \ldots, n$ e seus tempos de processamento por $p_i$, $i \in T$. A k-ésima ocorrência da tarefa $i$ é denotada por $< i, k >$, e o conjunto $I_k = \{< i, k > | \ i \in T\}$ é chamado de iteração $k$.

*Seja um escalonamento $S$ que atribui um tempo de início $t(i,k) \in \mathbb{R}$ para cada ocorrência $< i, k >$. O tempo de ciclo $\alpha(S)$ de $S$ é o comprimento de iteração médio, isto é:*

$$\alpha(S) = \lim_{k \longrightarrow \infty} \frac{máx_{i \in T}\{t(i,k) + p_i\}}{k},$$

[5]Qualquer seqüência de processamento das tarefas associadas a um recurso é factível.

*e o fluxo* $\tau(S)$ *de* $S$ *é definido como* $\tau(S) = 1/\alpha(S)$.

*Um escalonamento* $S$ *é periódico com período* $\alpha$ *se para cada tarefa genérica* $i$,

$$t(i,k) = t_i + \alpha * k, \tag{2.20}$$

*onde* $t_i = t(i,0)$.

*Note que o conjunto* $S_g = \{t_i \in \mathbb{R}_+ \mid i \in T\}$ *chamado escalonamento genérico, e o período* $\alpha$ *definem completamente um escalonamento periódico, se* $t_i$ *satisfizer a equação 2.20. Além disso,* $\alpha$ *é denominado tempo de ciclo de* $S$.

Seja $M = 1, \ldots, m$ o conjunto de máquinas especializadas[6] do JSR. Cada tarefa genérica é, a *priori*, alocada para um dado processador e $T_j$ denota um conjunto de tarefas genéricas atribuidas ao processador $j$. Inversamente $M_i$ é o processador atribuído a tarefa $i$.

Uma restrição conjuntiva genérica é uma quádrupla $(i,j,l,h)$, onde $i$ e $j$ são duas tarefas genéricas, $l$ é um número natural e $h$ é um inteiro. Um escalonamento cíclico $S$ satisfaz a restrição conjuntiva $(i,j,l,h)$ se para qualquer número natural $k$, $t(j,k+h) \geq t(i,k) + l$.

O conjunto de restrições conjuntivas de um JSR pode ser modelado por um grafo de precedência generalizada chamado de grafo conjuntivo. Os vértices desse grafo são as tarefas genéricas. A restrição $(i,j,l,h)$, é modelada por um arco de $i$ para $j$ e denotada por $e_{ij}$ com dois valores: seu comprimento $l$ e sua altura $h$, assim o conjunto dos arcos será $E = \{e_{ij} \mid i,j \in T\}$.

$G = (T,E)$ denota o grafo direcionado de restrições conjuntivas; $L : E \rightarrow N$ denota a função comprimento e $H : E \rightarrow Z$ denota a função altura nos arcos de $G$. Note que o comprimento de um arco é não-negativo, mas a altura pode ser negativa.

Para um melhor entendimento as notações e variáveis de decisão utilizadas na formulação de Hanen (1994) são listadas a seguir:

- $T$ : conjunto das operações do *job*;
- $z$ : comprimento do ciclo;
- $i,j,n$ : índices das operações associadas ao *job*;
- $e_{ij}$ : arco de $i$ para $j$ que estabelece a relação de precedência entre as operações ;
- $E$ : conjunto dos arcos $e_{ij}$;
- $\mathbb{M}$ : conjunto de máquinas;
- $m,s$ : índices das operações associadas às máquinas;

[6]Uma operação qualquer só pode ser processada na máquina associada a ela

- $m_i$ : única máquina capaz de processar a operação $i$;
- $T_s$ : conjunto de operações associadas a máquina $s$;
- $p_i$ : tempo de processamento da operação $i$;
- $L_{e_{ij}}$ : valor natural que pode ser escrito como $l_{ij}$ e que será definido a seguir:

$$l_{ij} = \begin{cases} p_i & \text{se } i \text{ precede } j \text{ no sequenciamento do } job \\ 0 & \text{caso contrário.} \end{cases}$$

- $H_{e_{ij}}$ : altura de recorrência entre as operações $i$ e $j$ e que pode ser escrita como $h_{ij}$;
- $K(i,j)$ : supondo $m_i = m_j$ esta variável está associada ao sequenciamento das tarefas $i$ e $j$ na máquina correspondente;
- $t_i$ : tempo de início de execução da operação $i$;
- $u_i$ : $t_i \,/\, z$;
- $\tau$ : fluxo do sistema equivalente a $1/z$;

O conjunto de restrições do problema de job-shop recorrente, que tem como objetivo maximizar o fluxo $\tau$ é o seguinte:

- para satisfazer a ordem de precedência especificada, utiliza-se:

$$u_j - u_i \geq \tau L_{e_{ij}} - H_{e_{ij}} \quad \forall e_{ij} \in E;$$

pode-se substituir a função $L$ de comprimento do arco, pelo tempo de processamento das tarefas $p_i$ (Hanen, 1994):

$$u_j - u_i \geq \tau p_{ij} - h_{ij} \quad \forall e_{ij} \in E; \tag{2.21}$$

- Para garantir a não sobreposição de tarefas em um máquina qualquer, utiliza-se:

$$u_j - u_i \geq \tau p_i - K(i,j) \quad \forall s \in \mathbb{M}, \quad \forall i,j \in T_s, \tag{2.22}$$

$$K(i,j) + K(j,i) = 1 \quad \forall s \in \mathbb{M}, \quad \forall i,j \in T_s, \tag{2.23}$$

$$K(i,j) \in \mathbb{Z}, \quad \forall s \in \mathbb{M}, \quad \forall i,j \in T_s, \tag{2.24}$$

- Os tempos de início das tarefas genéricas devem ser positivos e o fluxo estritamente positivo, assim tem-se:

$$u_i \geq 0 \quad \forall i, \tag{2.25}$$

$$\tau > 0, \tag{2.26}$$

Pela definições dadas nesta seção, sabe-se que $z = 1/\tau$ e $t = u/\tau = uz$, assim a formulação do JSR poderá ser reescrita de seguinte forma:

$$\begin{aligned}
&\min z \\
t_j - t_i &\geq p_i - h_{ij}z \quad \forall e_{ij} \in E, \\
t_j - t_i &\geq p_i - k(i,j)z \quad \forall s \in \mathbb{M}, \quad \forall i,j \in T_s, \\
k(i,j) + k(j,i) &= 1 \quad \forall s \in \mathbb{M}, \quad \forall i,j \in T_s, \\
t_i &\geq 0 \quad \forall i, \\
z &\geq 0, \\
k(i,j) &\in \mathbb{Z} \quad \forall s \in \mathbb{M}, \quad \forall i,j \in T_s,
\end{aligned} \tag{2.27}$$

Para visualizar melhor o problema discutido nesta seção , o mesmo exemplo apresentado na seção anterior será modelado usando-se a formulação proposta por Hanen (1994) e sua resolução será feita utilizando-se o software GAMS, que será melhor explicado no capítulo 4.

**Exemplo 2.2** : *A tabela a seguir mostra os tempos de processamento das operações de um job A obedecendo uma rota pré determinada* [7]*:*

| Job | Operação | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| A | (11,2) | (13,4) | (6,3) | (11,4) | (1,1) | (11,5) | (8,3) |

Tabela 2.2: Dados do exemplo 2.2

O objetivo é minimizar o período para a repetição das operações.

Como foi visto nesta seção, o conjunto de restrições conjuntivas de um JSR pode ser modelado por um grafo de precedência generalizada chamado de grafo conjuntivo, onde os vértices desse grafo serão as tarefas genéricas. Assim este problema também pode ser modelado através do grafo mostrado na figura 2.2.

[7] $(p_i,m_i)$ = (tempo de processamento da operação $i$, máquina associada a operação $i$)

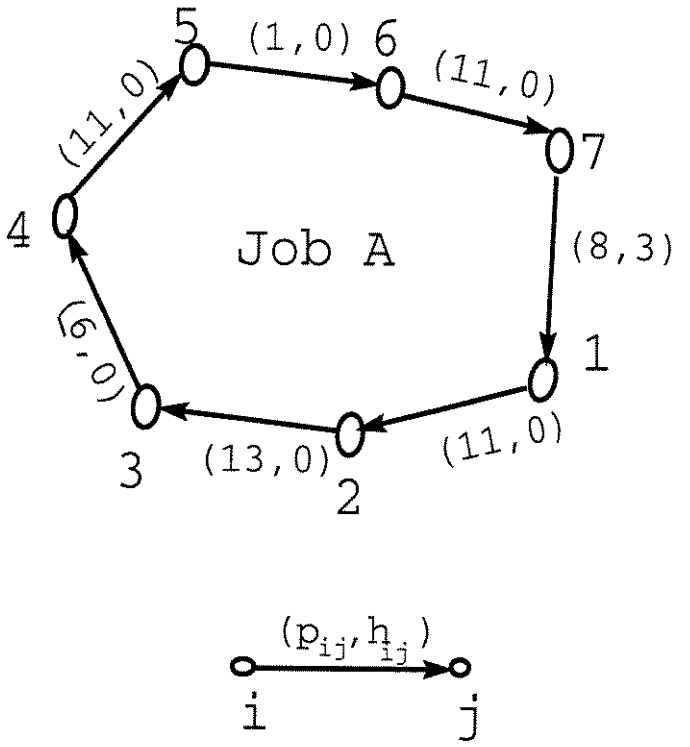



Figura 2.3: Exemplo de um Grafo de Job-Shop Recorrente

Para o modelo JSR é necessário estipular-se a *priori* a altura de recorrência entre a primeira e a última operação. Por isso, este exemplo foi resolvido para dois casos, cada caso com uma altura de recorrência diferente ( h=3 e h=4 ). No primeiro caso, com h=3, o comprimento de ciclo ótimo encontrado foi de 27,5 unidades de tempo (u.t.) e o escalonamento cíclico ótimo é dado pela figura 2.4. No segundo caso, pode-se notar pela figura 2.5 que ao estipular que a altura de recorrência entre a primeira e a última operação fosse igual a 4, encontrou-se o mesmo comprimento de ciclo ótimo e um escalonamento cíclico ótimo equivalente ao da seção anterior resolvido pela formulação de Rao (1992).

Pode-se observar ainda que a máquina 4 está sendo totalmente utilizada, por isso o comprimento do ciclo não poderá ser menor que 24 u.t..

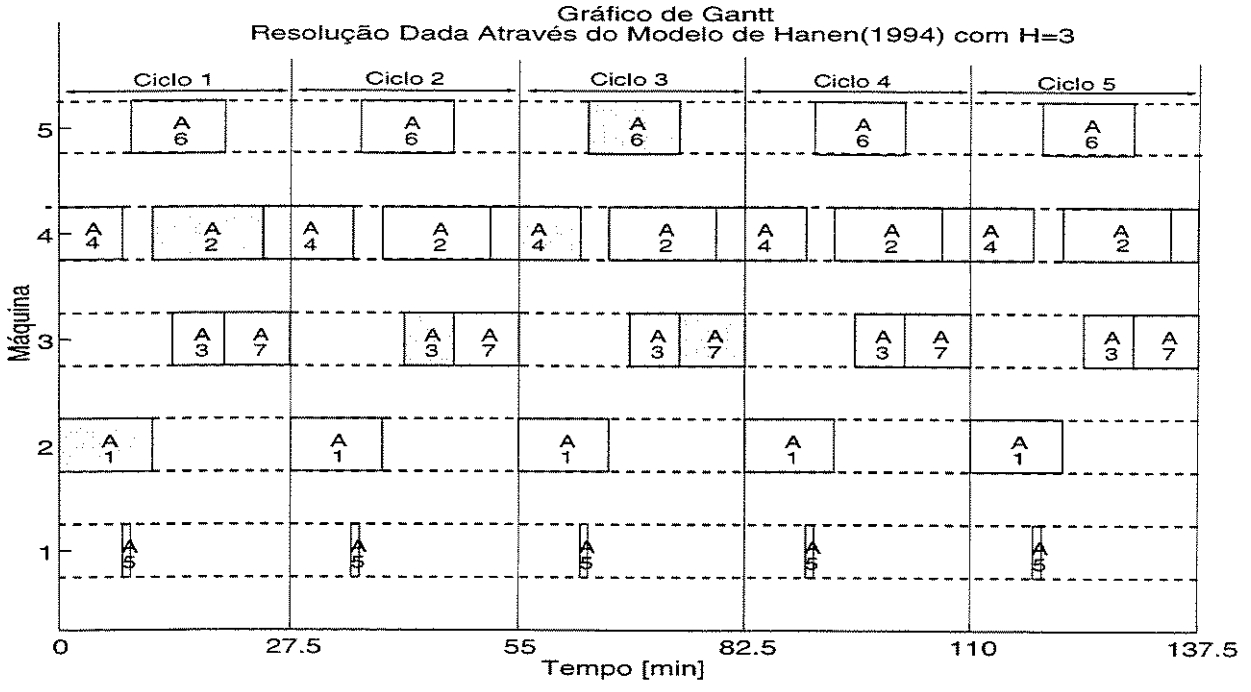



Figura 2.4: Exemplo de um Job-Shop Cíclico de Jobs Idênticos com H=3

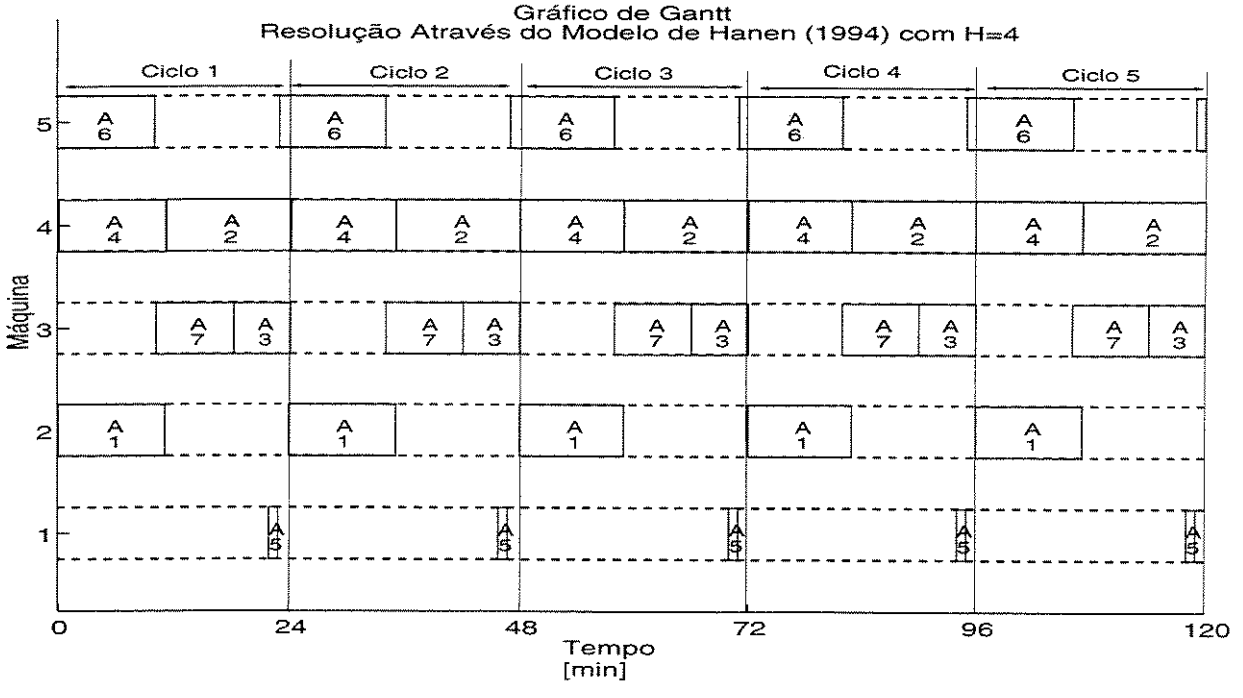



Figura 2.5: Exemplo de um Job-Shop Cíclico de Jobs Idênticos com H=4

## 2.3 Comparação entre os dois modelos

Como dito anteriormente o objetivo deste trabalho é modelar um problema de Job Shop Cíclico com Jobs Distintos, assim pesquisou-se, na literatura, modelagens que pudessem ser usadas para este fim. Os modelos ECJI (Rao, 1992) e JSR (Hanen, 1994) foram escolhidos por terem características semelhantes ao problema discutido neste trabalho, tais como: operações com tempo de processamento fixo, máquinas especializadas e recursos disjuntivos. O diagrama 2.6 mostra as principais características de cada modelagem.

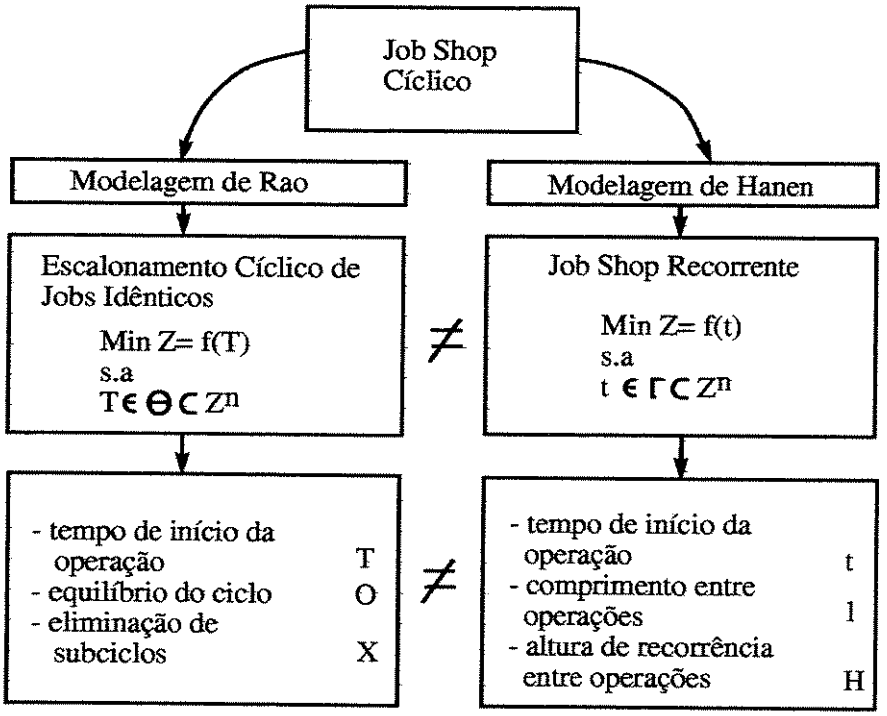



Figura 2.6: Características dos modelos

As principais diferenças entre os dois modelos observando-se as seções 2.1 e 2.2 e o diagrama 2.6 são listadas a seguir:

- tempo de início das operações medidos de forma diferente;
- a modelagem do JSR usa a altura de recorrência entre operações como um dos atributos das operações, porém este atributo não tem correspondência direta na modelagem do ECJI;
- a formulação do ECJI possui uma maior quantidade de equações que a do JSR;
- a formulação do JSR contem variáveis inteiras, enquanto a do ECJI possui apenas variáveis binárias.

Estes dois modelos não tratam do caso de *jobs* distintos, assim no próximo capítulo serão feitas modificações nos modelos para se adequarem a este caso.

## 2.4 Outras Modelagens

Alguns artigos na área de escalonamento cíclico, apesar de não serem usados diretamente neste trabalho, merecem uma atenção especial. Assim o objetivo desta seção é apresentar um breve resumo destes artigos.

Foi proposta por Dauscha *et al.* (1985), uma formulação para problemas de escalonamento periódico similar aos modelos de Hanen (1994) e de Rao (1992). Lidando com atividades de duração fixa sujeitas a restrições com uma distância de tempo mínimo entre seus tempos ordenados de ativação, ocasionou *ordens de precedência cíclica.* Como ferramenta matemática foram usados "tipos de seqüência cíclica", expressando-se o número de períodos necessários para cumprir qualquer conjunto cíclico dado de atividades em uma ordem pré-estabelecida. Depois derivou-se algumas propriedades interessantes desses tipos de seqüência cíclica, além de uma condição necessária e suficiente para garantir a existência de um escalonamento periódico. Um método para construir escalonamentos periódicos foi apresentado, e além disso, o problema global de achar um escalonamento cíclico admissível provou-se ser NP-completo e foi formulado como um problema de programação inteira mista com variáveis binárias. Uma aplicação para um problema simples relativo a um sistema de tráfego leve, operando periodicamente, foi minuciosamente estudado.

Song & Lee (1998) estudaram o problema de escalonamento para o caso geral de job-shops cíclicos com bloqueio, onde cada máquina possuía um *buffer* de capacidade finita. Assim desenvolveram modelos em Redes de Petri para os *shops*, analisando os casos de *buffers* sem capacidade e com capacidade finita e infinita, depois propuseram uma política de controle de *buffer* sequencial, fazendo com que os *jobs* fossem colocados no *buffer* de uma determinada máquina em uma seqüência especificada. Em seguida, mostraram que se o modelo de escalonamento de um *shop* cíclico com *buffers* finitos adotasse esta política de controle de *buffer*, este modelo poderia ser transformado em um de escalonamento de *shop* cíclico sem *buffer*, e ainda poderia ser modelado como um grafo marcado temporizado. Caracterizou-se, ainda, propriedades estruturais para a detecção de impasse (*deadlock*). E foi então apresentado um modelo em programação linear inteira mista para achar um escalonamento livre de impasses que minimizaria o tempo de ciclo.

# 3

# Problema de Job-Shop Cíclico de Jobs Distintos

Este capítulo trata de um problema de escalonamento cíclico com *jobs* distintos e fluxo reentrante, denominado aqui **Job-Shop Cíclico** .

Como qualquer problema de escalonamento cíclico, o Job-Shop Cíclico (JSC) se caracteriza por um conjunto de operações que são processadas infinitas vezes. Assume-se que o conjunto de todas as operações, pode ser particionado exaustivamente em sub-conjuntos disjuntos e mutuamente exclusivos chamados de *jobs* (Conway *et al.*,1967). Cada *job* consiste, então, de um conjunto de operações que devem ser realizadas em alguma ordem. Assume-se que os atributos das operações incluem a associação a uma única máquina ($m_{Ji}$), tempo de processamento fixo ($p_{Ji}$), somente um único sucessor no *job* ($s_{Ji}$), uma vez que a operação começou a ser processada não pode ser interrompida para execução de outra operação, e ainda, as operações são não reentrantes[1]. A k-ésima inicialização e execução de todas as operações de um *job* será tratada aqui como a k-ésima iteração do *job*.

Define-se neste trabalho que o tempo transcorrido entre a inicialização da primeira operação e o término da execução da última operação de um *job* será chamado de comprimento do *job*. O comprimento mínimo de um *job* será dado pela soma dos tempos de processamento de todas as operações deste *job*. Define-se também que o menor inteiro maior ou igual que a razão entre o comprimento do *job* e o comprimento do ciclo será tratado por largura do *job*. Assim tem-se:

$$largura = \left\lceil \frac{\text{comprimento do } job}{\text{comprimento do ciclo}} \right\rceil$$

Logicamente como o problema é periódico, a largura de um *job* é única, a partir do instante que se determina um escalonamento cíclico.

Como este trabalho lida com *jobs* distintos, com comprimentos mínimos distintos, o comprimento

[1] As operações não se sobrepõem.

e a largura dos *jobs* pode variar muito, assim é necessário haver um controle de largura. O exemplo a seguir ilustra melhor esta idéia.

**Exemplo 3.1** : *Sejam três jobs, A, B e C, com 5, 10 e 9 operações respectivamente, obedecendo uma rota pré determinada descrita pela seguinte tabela* [2]*:*

| Job | Operação | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| A | (11,1) | (6,3) | (15,1) | (2,4) | (11,1) | (.) | (.) | (.) | (.) | (.) |
| B | (12,1) | (11,5) | (9,3) | (3,4) | (10,3) | (12,1) | (15,5) | (8,3) | (3,2) | (14,5) |
| C | (6,3) | (13,5) | (6,3) | (3,4) | (9,3) | (1,4) | (4,2) | (15,1) | (4,2) | (.) |

Tabela 3.1: Dados do exemplo 3.1

Este problema será modelado usando-se a formulação ECJD que será vista mais adiante na seção 3.2 e sua resolução será feita utilizando-se o software GAMS, que será melhor explicado no capítulo 4.

O objetivo é minimizar o período para a repetição das operações.

O período ótimo sem o controle de largura é de 76 unidades de tempo e o escalonamento ótimo desse caso é mostrado através do gráfico de Gantt apresentado na fig. 3.1:

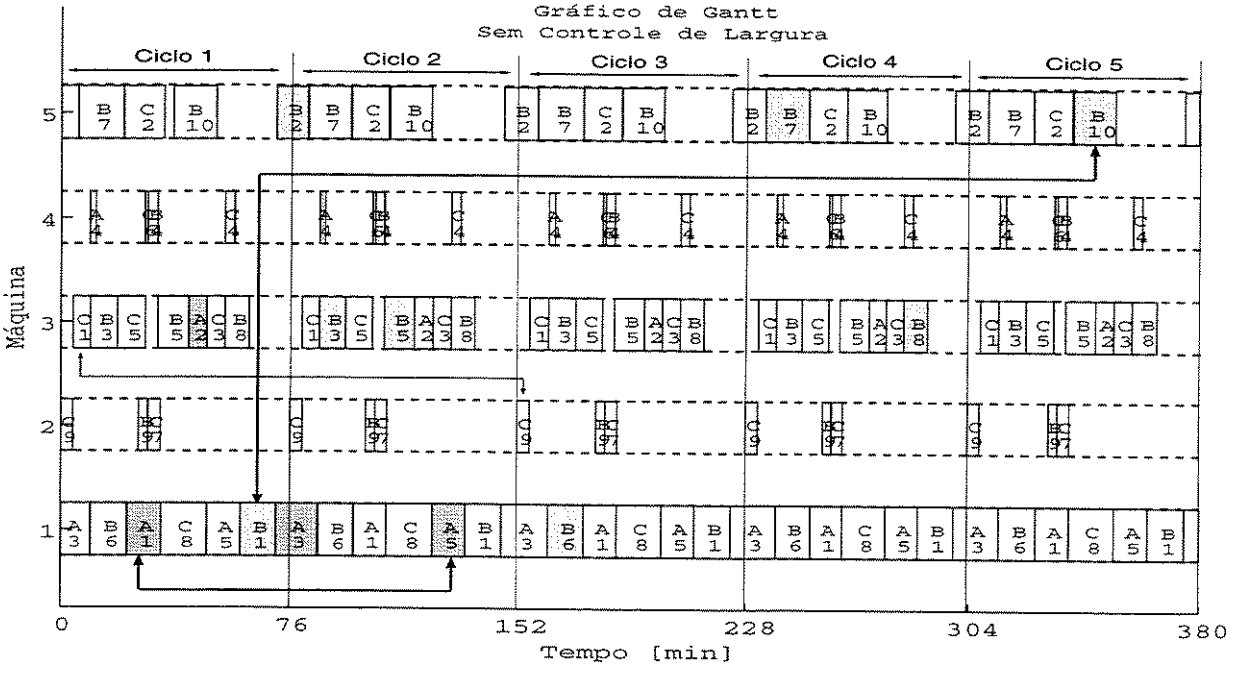



Figura 3.1: Exemplo de um Job-Shop Cíclico Sem Controle de Largura

[2]Onde $(p_i,m_i)$ = (tempo de processamento da operação $i$, máquina associada à operação $i$)

As setas indicam o comprimento dos *jobs* A e B, assim pode ser notado que qualquer iteração do *job* B permanece no sistema por um período bem maior que qualquer iteração do *job* A, o que poderia ser natural visto que B tem uma duração mínima de 97 u.t., enquanto A tem uma de 45 u.t.. Usando-se a restrição de controle de largura, para que cada instância de qualquer *job* tenha largura máxima igual a 2 ciclos, pode-se verificar na figura 3.2 que a cada 2 ciclos alguma iteração dos *jobs* A, B e C é finalizada, sendo que o período ótimo continua a ser de 76 unidades de tempo.

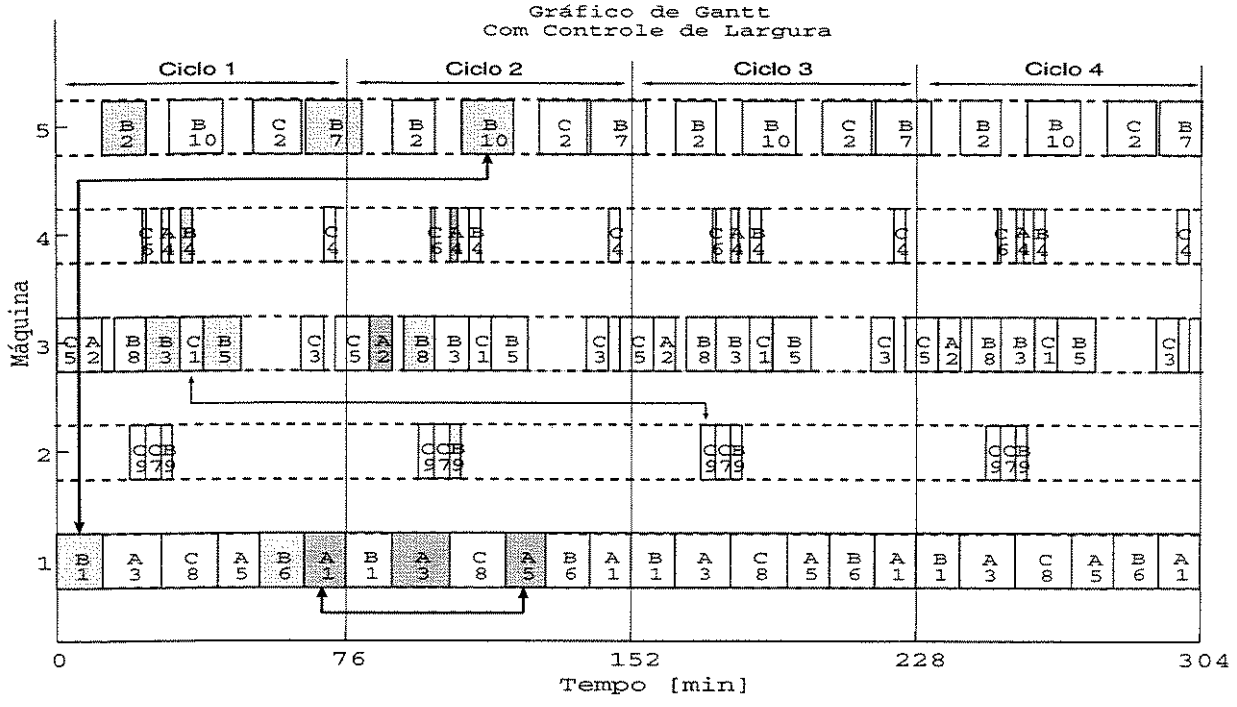



Figura 3.2: Exemplo de um Job-Shop Cíclico Com Largura $\leq 2$

Um atributo, que apresenta uma forte correspondência com a largura, é o da altura de recorrência entre operações, tanto em relação às instâncias de um único *job* (Hanen, 1994), quanto em relação aos *jobs* distintos. A altura de recorrência entre duas operações$i$ e $j$ é melhor descrita através da equação $t_{j,k+h} \geq t_{i,k} + p_i$, $h \in \mathbb{Z}$, para qualquer número natural $k$. Assim a h-ésima recorrência da k-ésima ocorrência da operação $j$ só pode ser inicializada após o término da k-ésima ocorrência de $i$. Usando esta definição , será estudado neste trabalho o comportamento dos modelos ECJI e JSR modificados, propondo-se alguns valores de altura de recorrência entre a primeira e a última operação de cada *job*, e também entre os *jobs* visando garantir o equilíbrio na produção dos *jobs* distintos.

Ainda pela definição , a altura de recorrência no *job* pode ser dada por um número fixo que limita a quantidade de instâncias de um mesmo *job* que podem ocorrer no mesmo ciclo. Assim, enquanto a largura limita superiormente a quantidade de ciclos utilizados para a finalização de cada iteração

de um *job*, a altura de recorrência de um *job* limita superiormente a quantidade de instâncias que podem ocorrer num mesmo período. A figura 3 ilustra melhor esta idéia, mostrando um exemplo com 3 *jobs* distintos e altura de recorrência igual a 2, e com uma solução factível. Como pode ser notado, cada iteração dos *jobs* 1, 2 e 3 leva dois ciclos ou menos para ser finalizada, isto corresponde a estabelecer-se uma largura igual ou menor a 2. Esta correspondência entre largura e altura torna possível a equivalência entre os modelos escalonamento cíclico de *jobs* distintos e job shop recorrente adaptado e que será vista mais adiante neste capítulo.

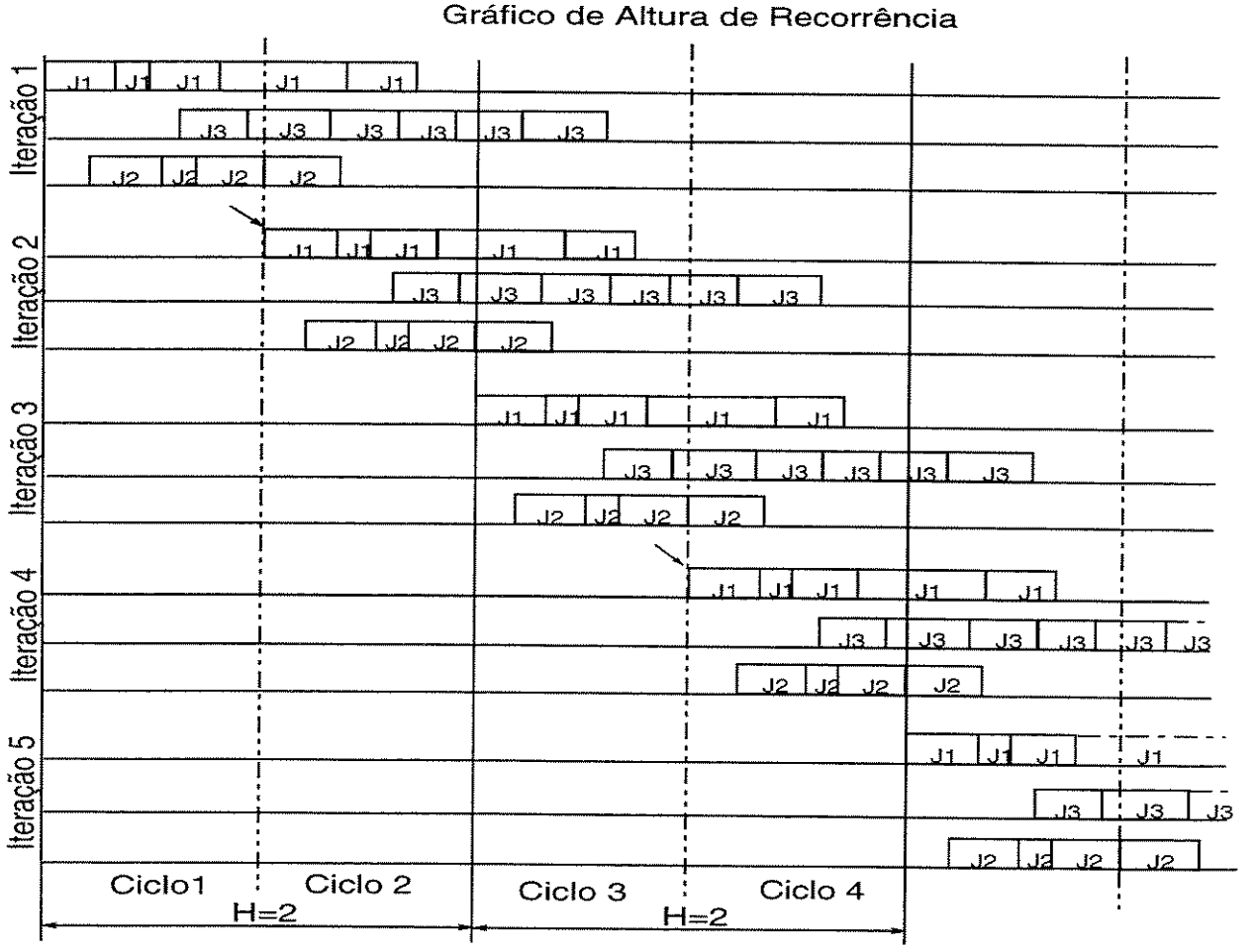



Figura 3.3: Exemplo de um Job-Shop Cíclico Com Altura de Recorrência igual a 2

Como foi visto no capítulo anterior o JSR é caracterizado por um conjunto de tarefas genéricas que são executadas um número muito grande de vezes e Hanen (1994) em seu estudo modelou um conjunto de restrições conjuntivas genéricas de um JSR através de um grafo de precedência generalizado chamado de grafo conjuntivo e através desta modelagem propôs um controle de fluxo, chamado por ela de altura

de recorrência. O JSR também foi modelado como um problema de programação linear inteira mista, onde para garantir a não sobreposição de operações na mesma máquina foi utilizada uma equação com variáveis inteiras. Esta restrição deixou a modelagem menos robusta que a proposta por Rao (1992).

Rao(1992) em seu estudo do ECJI modelou-o como um problema de programação inteira mista, sendo que as variáveis da formulação foram decompostas em 2 conjuntos; um conjunto combinatorial, chamado de estrutura de precedência cíclica, caracterizando todas as relações de precedência entre operações em um escalonamento cíclico, e um conjunto de variáveis contínuas de momento de produção, caracterizando os tempos de início das operações e o comprimento do ciclo. Esta estrutura de precedência cíclica não estabeleceu nenhum controle de fluxo, exceto o relativo a não sobreposição de operações em uma máquina, por isso será necessário acrescentar restrições para fazer o controle através da altura de recorrência e isto poderá ser realizado utilizando-se a largura do *job*.

Nenhuma das modelagens (Rao, 1992 e Hanen, 1994) tratou do caso de *jobs* distintos, por isso existe uma necessidade de estabelecer-se uma relação entre os *jobs*, que será definida aqui neste trabalho e será chamada de altura de recorrência entre *jobs*.

Assim o problema de job-shop cíclico de *jobs* distintos pode ser modelado através de adaptações feitas às formulações do problema de job-shop recorrente (Hanen, 1994) e do problema de escalonamento cíclico de *jobs* idênticos (Rao, 1992). Apesar destas mudanças a resolução de qualquer problema de job shop cíclico com *jobs* idênticos não será diferente da resolução feita pelo modelo não alterado, a não ser no caso do controle de largura e que pode ser verificado na seção 3.2. A equivalência destes dois novos modelos será mostrada no capítulo 6.

Estes dois novos modelos tem como objetivo minimizar o período $z$ e que equivale a maximizar o fluxo $\tau$, e serão apresentados em seguida.

## 3.1 Simbologia

### 3.1.1 Notação

Os seguintes símbolos serão utilizados nesta dissertação :

$\mathbb{M}$ – conjunto de máquinas;

$m, s$ – índice de máquina $1 \leq m \leq card(\mathbb{M})$;

$\mathbb{O}_m$ – conjunto de operações associadas a uma máquina;

$\mathbb{J}$ – conjunto de *jobs*;

$J, Q$ – índices relativos a *jobs* $1 \leq J, Q \leq card(\mathbb{J})$;

$\mathbb{O}_J$ – conjunto de operações de um *job* $J$;

$i, j, n$ – índices de operações associadas a um *job* $J$ $\quad 1 \leq i, j, n \leq card(J)$;

$m_{Ji}$ – única máquina capaz de processar a operação $i$ do *job* $J$;

$\sigma_{Ji}$ – sucessor imediato da operação $i$ do *job* $j$ na máquina $m_i$, onde $\sigma_{Ji} = Qj$;

$s_{Ji}$ – único sucessor da operação $i$ no *job* $J$;

$p_{Ji}$ – tempo de processamento da operação $i$ do *job* $J$.

$l_{Qj,Ji}$ – valor natural dado por:

$$l_{Ji,Qj} = \begin{cases} p_{Ji} & \text{se } J = Q \text{ e se } i \text{ precede } j \text{ no sequenciamento} \\ & \text{do } job\ ; \\ 1 & \text{se } Ji \text{ e } Qj \text{ representam a relação de altura} \\ & \text{de recorrência entre } jobs. \end{cases}$$

$p_{Q1,J1}$ – valor constante .

$h_{Ji}$ – altura de recorrência da operação $i$ do *job* $J$;

$$h_{Ji} = \begin{cases} \text{H} & \text{se } i\text{=card}(J),\ H \in \mathbb{N}; \\ 0 & \text{caso contrário.} \end{cases}$$

### 3.1.2 Constantes

$B$ – limite superior do ciclo;

$n$ – estimativa geral para a quantidade de operações associadas às máquinas.

### 3.1.3 Variáveis de Decisão

$t_{Ji}$ : tempo de início de processamento da operação $i$ do *job* $J$;

$T_{Ji}$ : $t_{Ji}$ mod $z$;

$x_{Ji,Qj}$ : variável binária que indica a seqüência das operações de uma mesma máquina;

$$x_{Ji,Qj} = \begin{cases} 1 & \text{se } m_{Ji} = m_{Qj} \text{ e } i \text{ precede imediatamente } j \text{ na máquina } (Qj = \sigma_{Ji}); \\ 0 & \text{caso contrário } (Qj \neq \sigma_{Ji}). \end{cases}$$

$O_{Ji}$ : indica a posição com relação ao ciclo, *cycle offset*, entre as operações $i$ e $s_i$ do *job* $J$;

$$O_{Ji} = \begin{cases} 0 & \text{se } s_i \text{ é inicializado no mesmo ciclo que a operação } i, \\ 1 & \text{se } s_i \text{ é inicializado no ciclo seguinte relativo ao início da operação } i, \\ 2 & \text{caso contrário.} \end{cases}$$

$L_{Ji}$ : indica a última operação a ser inicializada no ciclo em uma máquina;

$$L_{Ji} = \begin{cases} 1 & \text{se } Ji \text{ é a última operação da máquina } m_{Ji} \text{ no ciclo;} \\ 0 & \text{caso contrário.} \end{cases}$$

$k(Ji, Qj)$ : variável inteira;

## 3.2 Alterações nos modelos para satisfazer as especificações do problema

- **Job-Shop Recorrente Adaptado (JSRA)**

O job-shop recorrente adaptado pode ser representado como um grafo genérico, mostrado na figura 3.2, onde os vértices representam as operaçõesdo *job*, os arcos conjuntivos entre as operações do mesmo job representam a estrutura de precedência e os arcos disjuntivos entre os *jobs* representam a conservação da altura de recorrência, estes últimos tendo por finalidade estabelecer um equilíbrio na produção dos *jobs* ao longo do tempo, os valores sobre os arcos $(l_{ij}, h_{ij})$ representam respectivamente o comprimento entre operações e a altura de recorrência entre operações (3.1). A relação entre as operações de uma mesma máquina não serão mostradas neste grafo. Este grafo modela o problema descrito no exemplo 3.1.

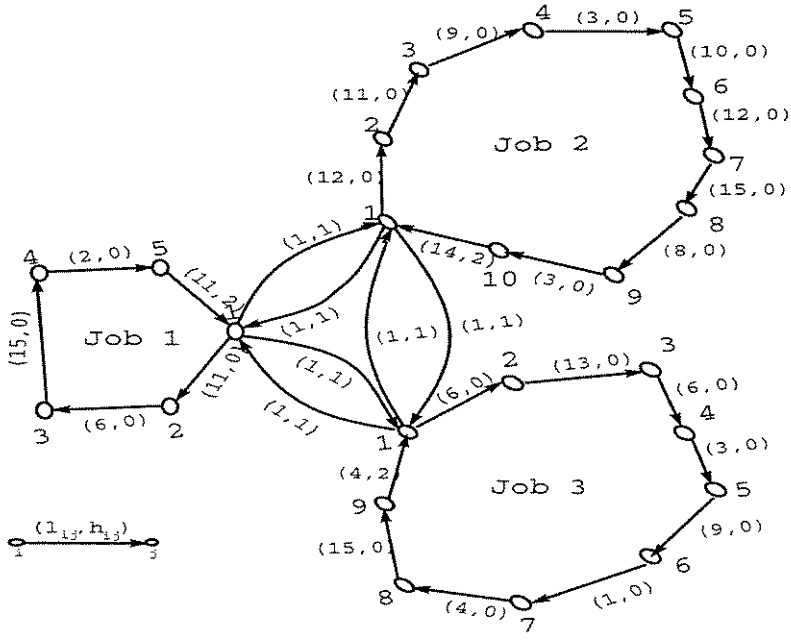



Figura 3.4: Exemplo de um Job-Shop Cíclico

O seguinte conjunto de restrições adicionais do modelo apresentado por Hanen (1994), descrito na Seção 2.2, é proposto para solucionar o Problema de Job-Shop Cíclico:

- Cada *job* consiste de um conjunto de operações que devem ser realizadas em alguma ordem. Assim estabelecida esta ordem pode-se sugerir um escalonamento cíclico, estabelecendo-se os tempos de inicialização de cada operação, de acordo com esta ordem de precedência; então com a periodicidade surge uma relação de pseudo-precedência entre a primeira e a última operação do *job*. Pode-se explicitar esta pseudo-precedência determinando que a k-ésima iteração da última operação do *job* J, seja finalizada até a inicialização da (k+h)-ésima iteração da primeira operação deste *job*, onde J é um *job* qualquer e $h$ é um número inteiro. Estas duas relações, precedência e pseudo-precedência, podem ser representadas pela seguinte equação:

$$t_{Jj} - t_{Ji} \geq L_{Ji,Jj} - h_{Ji}z \quad \forall i,j \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}.$$

E pela definição de $L_{Ji,Jj}$ pode-se fazer:

$$t_{Jj} - t_{Ji} \geq p_{Ji} - h_{Ji}z \quad \forall i,j \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}. \tag{3.1}$$

Esta relação de pseudo-precedência é chamada neste trabalho de altura de recorrência do *job*.

- O conjunto de operações associadas a uma determinada máquina obedece a um certo escalonamento parcial gerado na resolução do problema, porém é necessário garantir a não sobreposição de operações na máquina e para isso as seguintes restrições serão utilizadas:

$$t_{Qj} - t_{Ji} \geq p_{Ji} - k(Ji,Qj)z \quad \forall Ji,Qj \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{3.2}$$

$$k(Ji,Qj) + k(Qj,Ji) = 1 \quad \forall Ji,Qj \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{3.3}$$

$$k(Ji,Qj) \in \mathbb{Z} \quad \forall Ji,Qj \in \mathbb{O}_m, \quad \forall m \in \mathbb{M}. \tag{3.4}$$

- Para se ter um equilíbrio na produção dos *jobs* distintos ao longo do tempo, estipulou-se uma altura de recorrência entre as primeiras operações dos *jobs* distintos e que é mostrada a seguir:

$$t_{J1} - t_{Q1} \geq L_{Q1,J1} - z \quad \forall J,Q \in \mathbb{J}. \tag{3.5}$$

E pela definição de $L$ pode-se fazer:

$$t_{J1} - t_{Q1} \geq 1 - z \quad \forall J,Q \in \mathbb{J}. \tag{3.6}$$

$$O_{Ji} \in \mathbb{Z} \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}, \tag{3.23}$$

$$v_{Ji} \in \mathbb{N} \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}, \tag{3.24}$$

$$z \geq 0. \tag{3.25}$$

E o conjunto de restrições do ECJD será então:

$$\begin{aligned}
&\min Z \\
T_{s_{Ji}} - T_{Ji} &\geq p_{Ji} - O_{Ji} z \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}; \\
T_{\sigma_{Ji}} - T_{Ji} &\geq p_{Ji} - L_{Ji} z \quad \forall Ji \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}; \\
\sum_{Ji \in \mathbb{O}_s} L_{Ji} &= 1 \quad \forall s \in \mathbb{M}; \\
v_{Ji} - v_{Qj} + n x_{ij} &\leq (n-1) + n(L_{Ji} + L_{Qj}) \quad \forall Ji, Qj \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}; \\
\sum_{Ji \in \mathbb{O}_s} x_{Ji,Qj} &= 1 \quad \forall Qj \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}; \\
\sum_{Qj \in \mathbb{O}_s} x_{Ji,Qj} &= 1 \quad \forall Ji \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}; \\
T_{J1} - T_{Jn} &\geq p_{Jn} - O_{Jn} z \quad n = card(\mathbb{O}_J) \quad \forall J \in \mathbb{J}, \\
\sum_{i=1; i \in \mathbb{O}_J}^{n} O_{Ji} &\leq H \quad \forall J \in \mathbb{J}; \\
\sum_{i=1; i \in \mathbb{O}_J}^{n} O_{Ji} &\geq 1 \quad \forall J \in \mathbb{J}; \\
T_{J1} - T_{Q1} &\geq 1 - z \quad \forall J, Q \in \mathbb{J}; \\
T_{Ji} &\geq 0, \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}; \\
T_{Ji} &< z, \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}; \\
O_{Ji} &\leq 2, \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}; \\
O_{Ji} &\in \mathbb{Z} \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}; \\
v_{Ji} &\in \mathbb{N} \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}; \\
z &\geq 0.
\end{aligned} \tag{3.26}$$

Para uma melhor visualização será mostrado a seguir o mesmo exemplo 2.1 do Capítulo 2 comparando-se os resultados do modelos Rao (1992) e ECJD.

**Exemplo 3.2** : *Será produzido um produto contendo 7 partes que devem ser executadas obedecendo uma certa ordem. Uma vez que uma operação começou a ser executada não pode ser interrompida para a execução de outra operação. Cada parte do produto é associada a uma única máquina e seu tempo de processamento é fixo. O objetivo é minimizar o período para a repetição das tarefas.*

Os dados do problema são mostrados na tabela a seguir [3]:

| Job | Operação | | | | | | |
|---|---|---|---|---|---|---|---|
| | O1 | O2 | O3 | O4 | O5 | O6 | O7 |
| J1 | (11,2) | (13,4) | (6,3) | (11,4) | (1,1) | (11,5) | (8,3) |

Tabela 3.2: Dados do exemplo 3.2

Como foi visto na seção 2.1, Rao (1992) propôs um modelo para solucionar um problema de escalonamento cíclico de *jobs* idênticos, e que corresponde ao problema de escalonamento cíclico de um único *job*. E com as mudanças propostas nesta seção, o modelo, para este caso, ficou mais restritivo, devido ao controle da largura do *job*. A figura 3.5 mostra o gráfico de Gantt do problema do exemplo 2.1 resolvido através do modelo proposto por Rao (1992) e a figura 3.6 este mesmo exemplo resolvido pelo modelo ECJD com o limite superior da largura do *job* igual a três, e que equivale a ter uma altura de recorrência igual a três. Note-se que o modelo ECJD foi mais restritivo, neste caso. Porém, controlando melhor o fluxo, é possível diminuir o tamanho do ciclo relaxando-se o limite superior da largura, isto é, fazendo-se a altura de recorrência ser igual a quatro, como mostra a figura 3.7. Dessa forma o modelo ECJD encontrou o mesmo escalonamento ótimo encontrado pelo modelo proposto por Rao (1992).

[3] $(p_i, m_i)$ = (tempo de processamento da operação $i$, máquina associada a operação $i$)

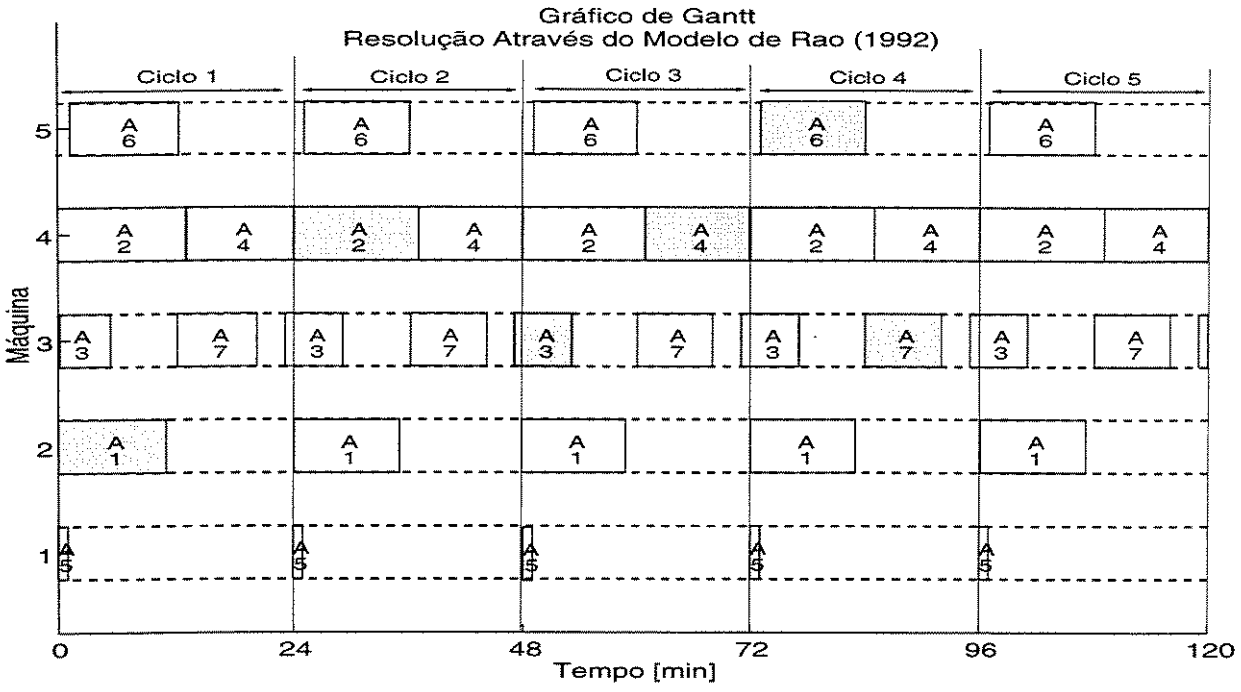



Figura 3.5: Escalonamento Cíclico de *Jobs* Idênticos Dado Pelo Modelo de Rao

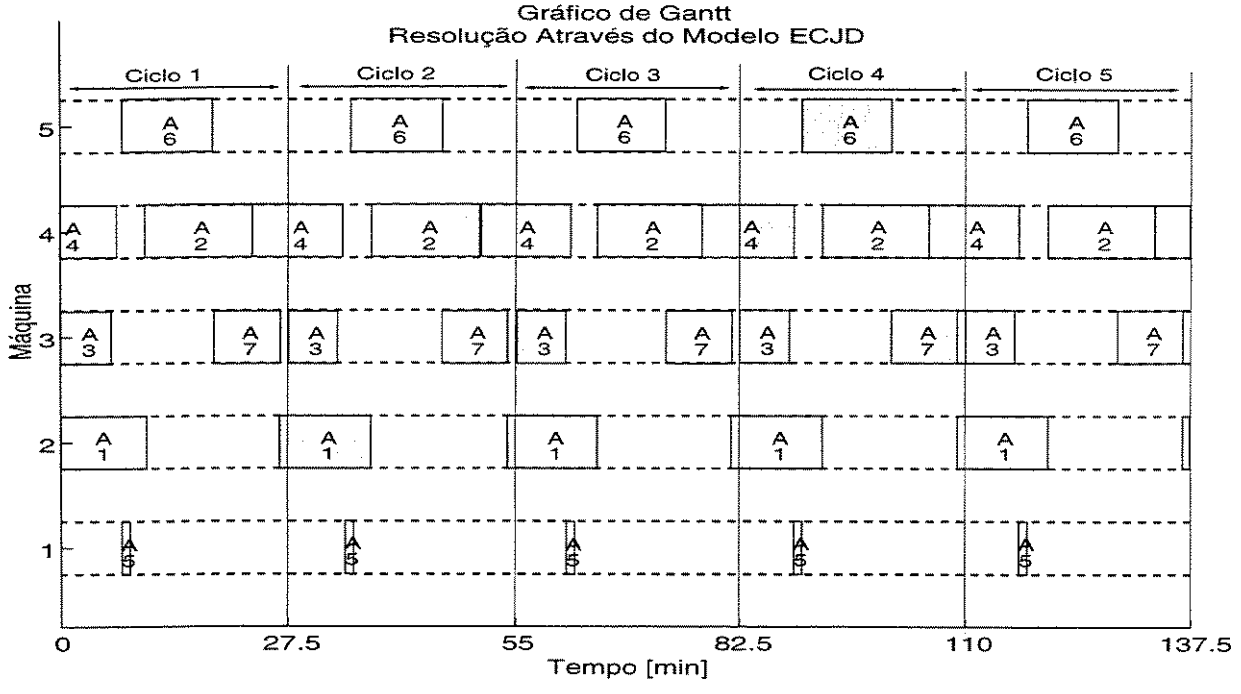



Figura 3.6: Escalonamento Cíclico de *Jobs* Idênticos Dado Pelo Modelo ECJD com H=3

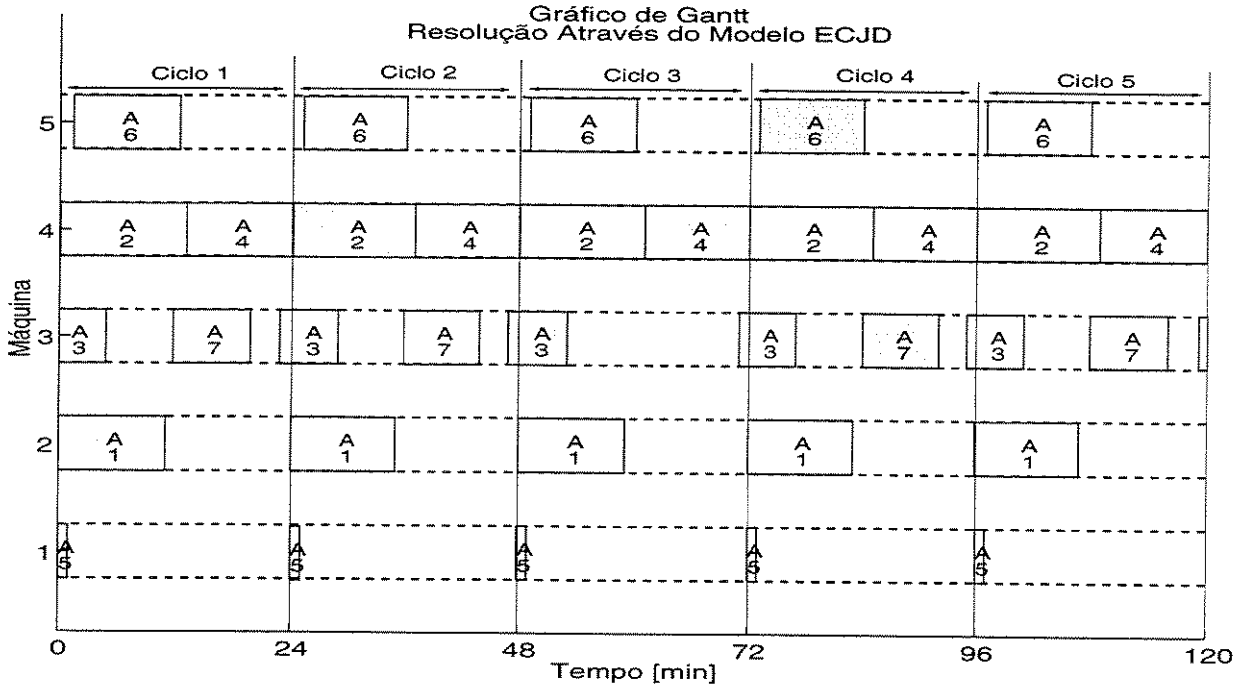



Figura 3.7: Escalonamento Cíclico de *Jobs* Idênticos Dado Pelo Modelo ECJD com H=4

## 3.3 Comparação entre os dois modelos

Como mostrado na seção 2.3 os modelos ECJI (Rao, 1992) e JSR (Hanen, 1994) têem características semelhantes ao problema discutido neste trabalho, tais como: operações com tempo de processamento fixo, máquinas especializadas e recursos disjuntivos. A figura 3.8 mostra um diagrama das principais características de cada modelagem e das modelagens adaptadas.

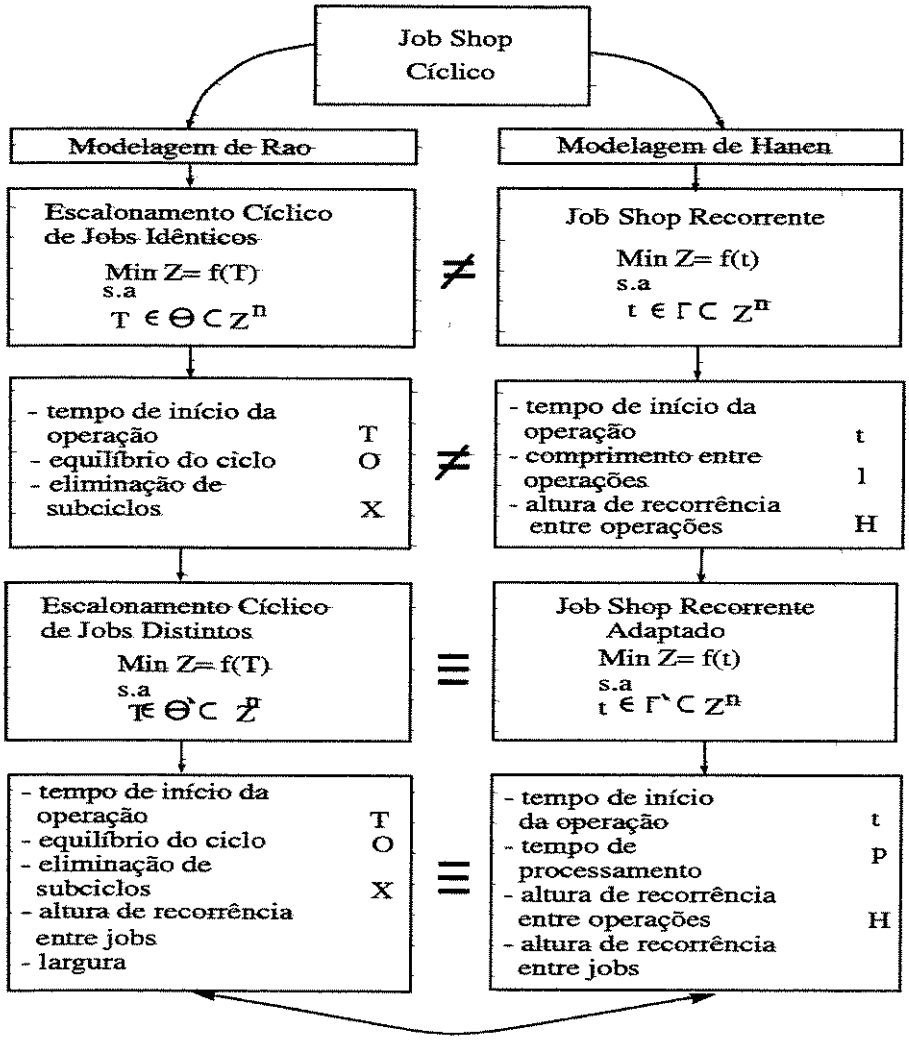



Figura 3.8: Características dos modelos

As principais diferenças entre os dois modelos foram listadas na seção 2.3 e através das modificações propostas neste capítulo, para eliminar essas diferenças, conseguiu-se a equivalência entre os modelos, mostrada no Cap. 6. Porém os modelos conservam as diferenças nos tipos das variáveis utilizadas (binária e inteira), e no próximo capítulo será possível perceber a influência dessas variáveis na resolução de exemplos gerados aleatoriamente.

# 4

# Resultados Experimentais

## 4.1 Geração Aleatória de Instâncias

Para se verificar a equivalência entre os modelos ECJD e JSRA foram gerados 160 exemplos de pequeno porte, para serem resolvidos por estes dois modelos, através do Sistema Geral de Modelagem Algébrica, **GAMS**, que é uma linguagem de alto nível para a formulação de modelos de Pesquisa Operacional. O *solver* OSL acoplado ao GAMS resolve problemas de programação linear inteira mista através da técnica de *Branch & Bound*. Para a execução deste *software* foi utilizado um microcomputador AMD-K6 II 333Mhz – 32 Mbytes de RAM.

Os exemplos foram divididos em grupos de um, de três e de cinco *jobs*, cada grupo com 40 exemplos. Estipulou-se que haveria três tipos de operações, as de curta, as de média e as de longa duração. Definiu-se que as máquinas seriam especializadas, isto é, cada máquina seria capaz de processar apenas um tipo de operação, cada tipo tendo um tempo mínimo e um máximo de processamento definidos a seguir:

- curta duração – 1 a 5 u.t.;
- média duração – 6 a 10 u.t.;
- longa duração – 11 a 15 u.t..

Definiu-se também um número máximo de máquinas igual a cinco.

No primeiro grupo, o de um *job*, cada exemplo é gerado contendo no mínimo 5 e no máximo 15 operações; assim primeiro é escolhida aleatoriamente a quantidade de operações do *job*, depois é associada, aleatoriamente, a cada operação, $i$, uma máquina, $m_i$, dessa forma dependendo da especialidade da máquina, gera-se aleatoriamente o tempo de processamento, $p_i$. Neste trabalho ficou determinado que a qualquer operação $j$, que segue imediatamente uma operação $i$, não pode ser associada a mesma máquina $m_i$, isto é, $m_i \neq m_j$.

Nos grupos de três e cinco *jobs*, a única diferença na geração dos exemplos é que cada *job* deve conter no mínimo 5 e no máximo 10 operações.

Na geração aleatória dos exemplos usou-se sempre uma distribuição uniforme.

**Observação:** Dada a grande complexidade da utilização do método de *Branch & Bound* para a resolução de um problema em programação inteira mista, foram acrescentadas inequações ao modelo, de forma a explicitar algumas propriedades (Apêndice C). Estas inequações não irão restringir o problema, mas lhe proporcionão um limite inferior melhor, fazendo com que a resolução do problema seja mais rápida.

## 4.2 Resultados

Um dos atributos do escalonamento cíclico de *jobs* distintos, é a altura de recorrência, assim para uma melhor compreensão dos modelos ECJD e JSRA, em cada exemplo foram estudados 3 tipos de altura de recorrência, gerando então 3 problemas diferentes para cada modelo, assim dados:

120 exemplos x 3 tipos de altura de recorrência x 2 modelos = 720 problemas

Tem-se então 720 problemas a serem resolvidos pelo GAMS/OSL, o que torna impraticável querer-se encontrar todas as soluções ótimas. Devido a natureza combinatória de um problema linear misto, a sua resolução pode demandar um tempo grande, assim ao se utilizar o programa GAMS/OSL, estipulou-se uma quantidade máxima de 1.500.000 iterações para se tentar achar a solução ótima de cada problema.

Os problemas estão divididos em 9 grupos, cada grupo contendo 80 problemas correspondendo aos 40 exemplos de um dos grupos de *jobs*, associados a uma única altura de recorrência e resolvidos pelos 2 modelos. Assim os resultados podem ser divididos em 9 partes e como se quer mostrar a equivalência dos modelos ECJD e JSRA, é necessário comparar os problemas que equivalem ao mesmo exemplo e a mesma altura de recorrência. Mas como foi estipulado um limite no número de iterações então serão feitos três tipos de avaliação .

- os dois modelos encontraram uma solução ótima para um mesmo problema;
- apenas um dos modelos encontrou uma solução ótima para um mesmo problema;
- nenhum dos dois modelos encontrou uma solução ótima para um mesmo problema.

A seguir serão mostrados os resultados separados em 4 tabelas, cada tabela correspondendo a um dos grupos:

Grupo I – corresponde a todos os problemas com um único *job*;

Grupo II – corresponde aos de três *jobs*;

Grupo III – corresponde aos de cinco *jobs*.

Em cada tabela serão mostrados os seguintes resultados:

A – corresponde a quantidade de problemas resolvidos por ambos o modelos em que foi encontrado o mesmo valor ótimo;

B –

ECJD – Quantidade de problemas resolvidos apenas por este modelo;

JSRA – Quantidade de problemas resolvidos apenas por este modelo;

ECJD $\leftrightarrow$ JSRA – Quantidade de problemas resolvidos por um dos modelos usando-se o escalonamento ótimo encontrado pelo outro modelo;

C – corresponde a quantidade de problemas não resolvidos por nenhum dos modelos;

D – corresponde a quantidade de problemas resolvidos por ambos o modelos em que não foi encontrado o mesmo valor ótimo.

Depois serão apresentados gráficos onde mostram-se os resultados dos modelos através de dois tipos de relação , fator de aumento e freqüência onde:

- O fator de aumento faz a relação entre o comprimento do *job*, no escalonamento cíclico ótimo calculado pelo modelo, e o comprimento mínimo do *job*, que é a soma de todos os tempos de processamento deste *job*. Portanto, o fator de aumento mostra o quanto a duração de um *job* foi estendida para satisfazer as relações entre os *jobs* do exemplo.

- A freqüência faz a relação entre o comprimento do *job* e o tempo de ciclo ótimo calculado através do modelo. Portanto, a freqüência mostra a quantidade média de iterações de um mesmo job que podem estar acontecendo ao mesmo tempo.

| Grupo I | | | | | | | |
|---|---|---|---|---|---|---|---|
| Altura | A | B | | | C | D | Total |
| | | ECJD | JSRA | ECJD ↔ JSRA | | | |
| 1 | 40 | . | . | . | . | . | 40 |
| 2 | 40 | . | . | . | . | . | 40 |
| 3 | 39 | . | 1 | 1 | . | . | 40 |

Tabela 4.1: Resultados dos experimentos de 1 Job

| Grupo II | | | | | | | |
|---|---|---|---|---|---|---|---|
| Altura | A | B | | | C | D | Total |
| | | ECJD | JSRA | ECJD ↔ JSRA | | | |
| 1 | 13 | 6 | 2 | 8 | 19 | . | 40 |
| 2 | 23 | 14 | . | 14 | 3 | . | 40 |
| 3 | 26 | 14 | . | 14 | . | . | 40 |

Tabela 4.2: Resultados dos experimentos de 3 Jobs

| Grupo III | | | | | | | |
|---|---|---|---|---|---|---|---|
| Altura | A | B | | | C | D | Total |
| | | ECJD | JSRA | ECJD ↔ JSRA | | | |
| 1 | . | . | 3 | 3 | 37 | . | 40 |
| 2 | 1 | 9 | 3 | 12 | 26 | . | 40 |
| 3 | 2 | 19 | 2 | 21 | 17 | . | 40 |

Tabela 4.3: Resultados dos experimentos de 5 Jobs

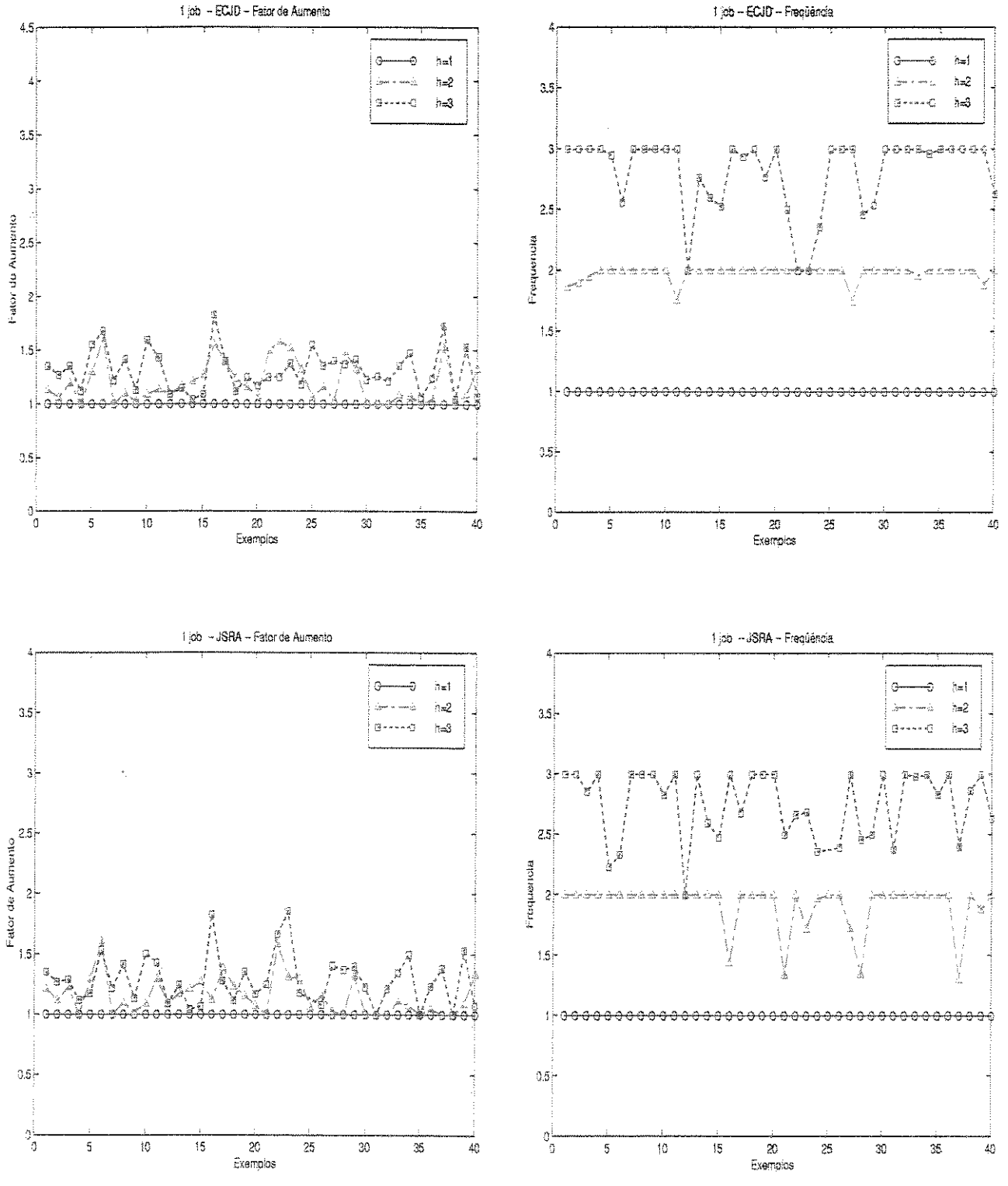



Figura 4.1: Resultados dos Experimentos - 1 Job

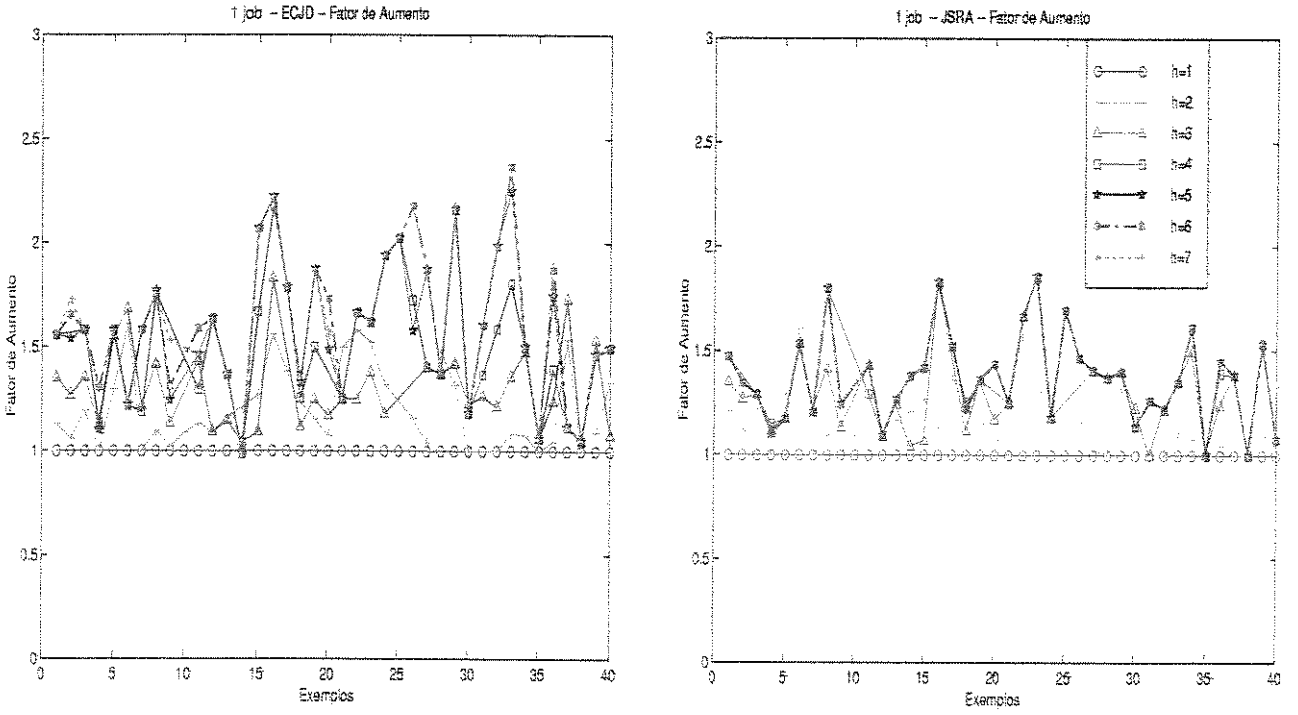



Figura 4.2: Comparação do fator de aumento

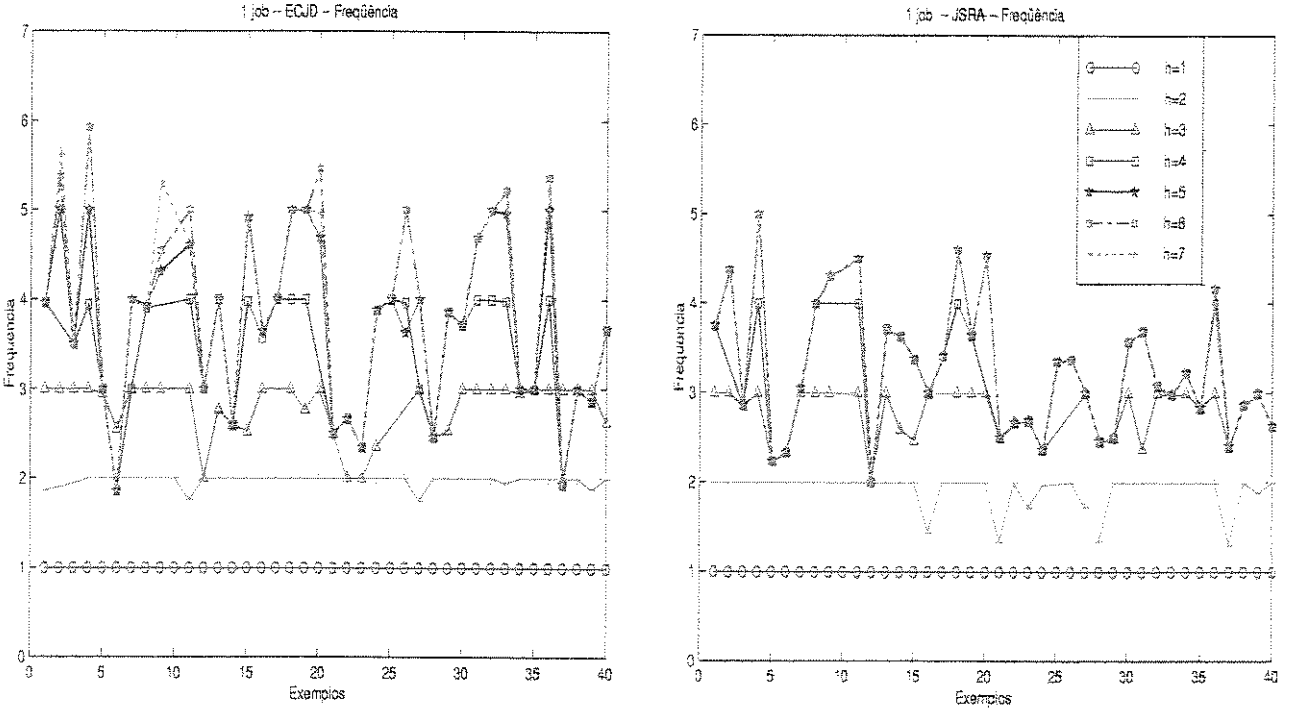



Figura 4.3: Comparação da freqüência

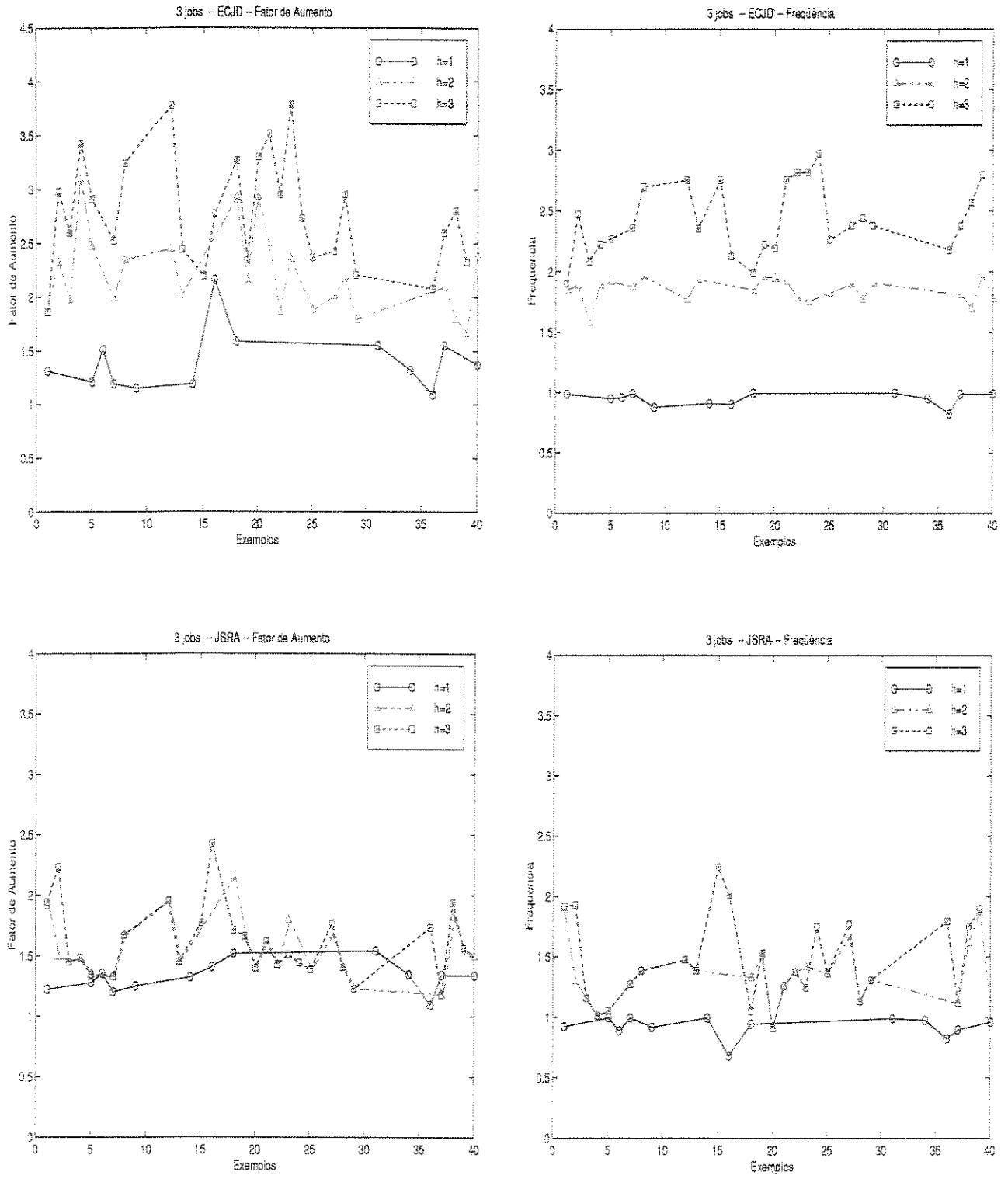



Figura 4.4: Resultados dos Experimentos - 3 Jobs

## 4.3 Análise dos Resultados

Na tabela 4.2 foram mostrados os resultados dos 120 exemplos gerados para o caso de um único *job*, e o que pôde ser verificado é que os dois modelos resolveram todos estes exemplos, a exceçao de um, chegando ao mesmo valor de ciclo ótimo. Este grupo de problemas difere dos demais no tocante à restrição de altura de recorrência entre *jobs*, que aqui é desnecessária, assim neste grupo é possível avaliar o efeito da restrição de altura de recorrência no *job*.

Este efeito pode ser melhor verificado nos gráficos da figura 4.1, assim os dois modelos tendem a aumentar o comprimento do *job*, à medida em que se aumenta o valor de $h$. Porém observando-se os gráficos relativo a freqüência, percebe-se que em alguns exemplos a 'altura de recorrência real' do *job* foi menor que a especificada no exemplo. Este é o caso do exemplo 28, que sendo resolvido pelos dois modelos teve uma freqüência menor que a altura de recorrência, e observando também o exemplo 12, percebe-se que a freqüência encontrada na resolução do exemplo com h=2 e com h=3 é a mesma, nos dois modelos.

Estas variações sugerem que existe um limite para que a altura de recorrência influencie na busca de um escalonamento cíclico. Para comprovar isto os 40 exemplos foram resolvidos novamente, mas para alturas de recorrência maiores, isto é, $h = 4$, $h = 5$, $h = 6$ e $h = 7$. Pode-se notar pela figura 4.3 que realmente existe uma altura de recorrência limitante. Para os exemplos resolvidos pelo ECJD o limitante assume o valor 6, enquanto nos resolvidos pelo JSRA este valor é 5.

Observando-se os gráficos de fator de aumento, tanto na figura 4.1 quanto na figura 4.2, verifica-se que o modelo ECJD aumenta mais o comprimento do *job* do que o JSRA, apesar do tamanho do ciclo ser o mesmo.

O caso de escalonamento cíclico de um único *job* pode ser chamado de escalonamento cíclico de *jobs* idênticos, e sua resolução pelo JSRA corresponde exatamente à resolução pelo modelo proposto por Hanen (1994). Porém a resolução através do ECJD difere da do modelo proposto por Rao (1992) apenas pelo controle de largura, conforme mostrado na seção 3.2, mesmo assim, pelos gráficos da figura 4.1, pode-se perceber a equivalência entre os modelos propostos por Hanen (1994) e Rao (1992) com controle de largura.

Nas tabelas 4.2 e 4.2 foram mostrados os resultados dos 120 exemplos gerados para os casos de três e de cinco *jobs* distintos, respectivamente, e o que se pôde constatar é que, com o aumento do tamanho do problema, a quantidade de problemas resolvidos por ambos os modelos diminuiu, principalmente em relação ao problema com altura de recorrência igual a 1. Porém é interessante observar que todos

os problemas que foram resolvidos apenas por um dos modelos inicialmente, foram solucionados pelo outro modelo utilizando-se o escalonamento ótimo encontrado por este, o que demonstra a equivalência destas soluções.

Os gráficos da figura 4.4 mostram a interação entre os *jobs* e que pode fazer com que um *job* estenda muito o seu comprimento para satisfazer a altura de recorrência, tanto relativo ao *job* quanto entre os *jobs*, assim observando-se os gráficos de fator de aumento, percebe-se que quanto maior o $h$ maior o comprimento médio do conjunto de *jobs* dos exemplos.

Os gráficos de freqüência, mostram também que na maioria dos exemplos a 'altura de recorrência real' dos *jobs* é menor que a altura de recorrência estipulada. Porém devido ao tamanho dos problemas, o modelo JSRA encontrou uma dificuldade maior para solucionar os exemplos em menos de 1.500.000 iterações, enquanto no modelo ECJD conseguiu-se solucionar mais da metade dos exemplos. Este fato está fortemente relacionado com a utilização de variáveis inteiras na formulação do JSRA e de variáveis binárias na do ECJD, e que faz com que a árvore de busca abra muitos mais nós para resolver o problema através do JSRA (e mantenha estes nós ativos), do que para resolver através do ECJD.

Não foram feitos gráficos para o caso de 5 *jobs* devido a quantidade menor de resultados ótimos conseguidos decorrente do aumento do tamanho do problema.

A necessidade de arbitrar valores para as constantes $B$ e $n$ no modelo ECJD, fez com que as resoluções dos problemas, através do GAMS/OSL, dependessem destes valores. Assim para o mesmo problema poder-se-ia ou conseguir uma solução em poucas iterações ou não conseguir solucionar o problema nem em dez milhões de iterações. Mas como não foi possível realizar um estudo mais aprofundado a esse respeito, contou-se, um pouco com a sorte, um pouco com o bom senso, para se encontrar as soluções dos problemas mais complexos. É preciso observar ainda que, a utilização de uma restrição de não existência de sub-ciclos diferente da sugerida por Rao (1992), não comprometeu os resultados dos problemas.

Na resolução do modelo JSRA, como foi observado anteriormente, na maioria dos problemas a quantidade de nós abertos na árvore de busca foi maior que a do ECJD, isto porque o JSRA usa variáveis inteiras, enquanto o ECJD usa variáveis binárias. E por causa dessa diferença o JSRA sempre conseguiu uma solução factível para o problema em poucas iterações, porém alguns exemplos não conseguiram atingir o limite de iterações, por limitações na capacidade da memória do computador.

No caso de um único *job*, o tempo médio de resolução foi de um minuto e apenas um único exemplo do modelo JSRA não obteve solução ótima. Este exemplo demorou 4 horas para completar o limite máximo de iterações.

No caso de três *jobs*, os exemplos do modelo ECJD que não obtiveram solução ótima tiveram uma duração média de 4 horas para completar o limite máximo de iterações e os exemplos do modelo JSRA, uma duração média de 6 horas.

No caso de cinco *jobs*, os exemplos do modelo ECJD que não obtiveram solução ótima tiveram uma duração média de 12 horas para completar o limite máximo de iterações e os exemplos do modelo JSRA, uma duração média de 18 horas.

O maior interesse desses experimentos era verificar a equivalência dos modelos, por isso, foram resolvidos problemas de pequeno porte através de um método exato, o **Branch & Bound**, através da utilização do programa GAMS/OSL.

# 5

# Estudo de Caso

Neste capítulo serão mostrados dois exemplos do problema de escalonamento cíclico, a fim de uma melhor compreensão e visualização do problema. Considera-se um *job* como sendo um conjunto de operações que devem ser realizadas em alguma ordem.

O primeiro exemplo, extraído do artigo de Groth & Santos-Mendes (1997), tratará de um caso de três *jobs*, onde os conjuntos de operações dos jobs tem a mesma quantidade de elementos (operações); e no segundo exemplo ser tratado um caso de três *jobs*, porém com os conjuntos de operações contendo quantidades de elementos diferentes.

**Exemplo 5.1** : *Serão produzidos três produtos, cada produto contendo seis partes que devem ser executadas obedecendo uma certa ordem. Cada parte do produto é associada a uma única máquina e seu tempo de processamento é fixo. Pode-se associar: produto a job, e parte do produto a operações. Cada máquina processará apenas uma única operação de cada job, portanto a cada máquina serão associadas exatamente três operações. Uma vez que uma operação começou a ser executada não pode ser interrompida para a execução de outra operação na máquina.*

Os dados do problema são mostrados na tabela a seguir [1]:

| Job | Operação | | | | | |
|---|---|---|---|---|---|---|
| | O1 | O2 | O3 | O4 | O5 | O6 |
| J1 | (3,4) | (8,2) | (9,5) | (10,3) | (8,6) | (8,1) |
| J2 | (7,5) | (9,6) | (10,2) | (2,3) | (4,4) | (5,1) |
| J3 | (9,4) | (10,6) | (3,2) | (6,3) | (2,5) | (10,1) |

Tabela 5.1: Dados do exemplo 5.1

[1] $(p_i,m_i)$ = (tempo de processamento da operação $i$, máquina associada à operação $i$)

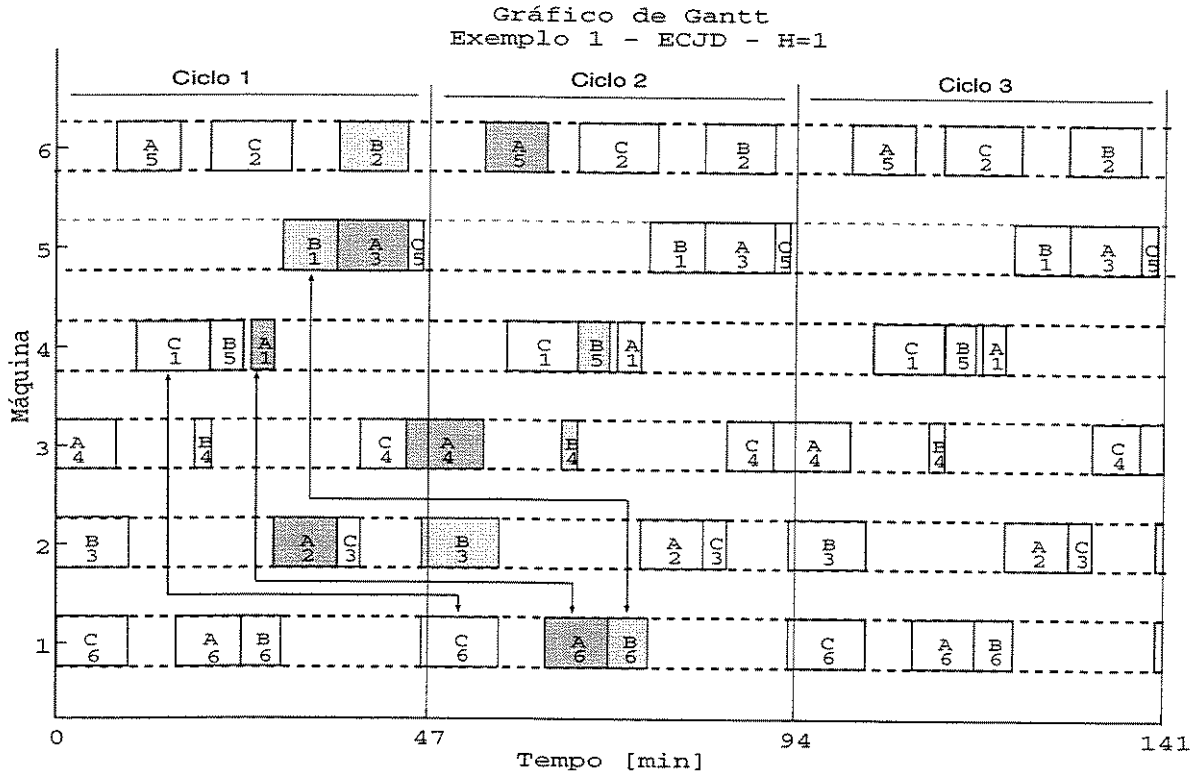
O objetivo é maximizar o fluxo do sistema, ou respectivamente, minimizar o período para a repetição das tarefas.

O problema será resolvido pelo programa **GAMS**, de duas maneiras, uma utilizando o modelo JSRA e a outra o ECJD. Os resultados serão mostrados em forma de carta de Gantt, geradas pelo software MatLab através dos arquivos de saída do **GAMS**. Nas cartas, as linhas verticais representam o ciclo ótimo, as linhas pontilhadas horizontais as máquinas e as setas mostram a duração de uma instância de cada *job* distinto.

É necessário ressaltar que as cartas de Gantt mostram apenas alguns períodos de tempo, assumindo-se que a solução obtida estende-se por todo o domínio do tempo.

O maior interesse deste trabalho é encontrar o estado regular ótimo do sistema de acordo com as restrições físicas e de fluxo. Pode-se verificar pela figura 5.1 que ao estabelecer-se uma altura de recorrência igual a 1 o comprimento do ciclo é maior que os outros casos, com $h = 2$ e $h = 3$ mostrados nas figuras 5.2 e 5.3 respectivamente, porêm o comprimento dos *jobs* será menor. Assim, em estado regular, quanto menor a altura de recorrência menor o tempo de armazenamento dos produtos intermediários.

Em cada ciclo existe sempre um exemplar de todas as operações, então a cada ciclo existe uma instância de cada *job* começando e outra terminando. Assim, tendo-se uma altura de recorrência maior, o comprimento do ciclo $Z$ pode diminuir (isto acontece apenas se todas as máquinas ainda tiverem tempo ocioso) e com isso em estado regular existirá uma maior quantidade de produtos finalizados ao longo do tempo. Isto pode ser observado nas cartas de Gantt subseqüentes.



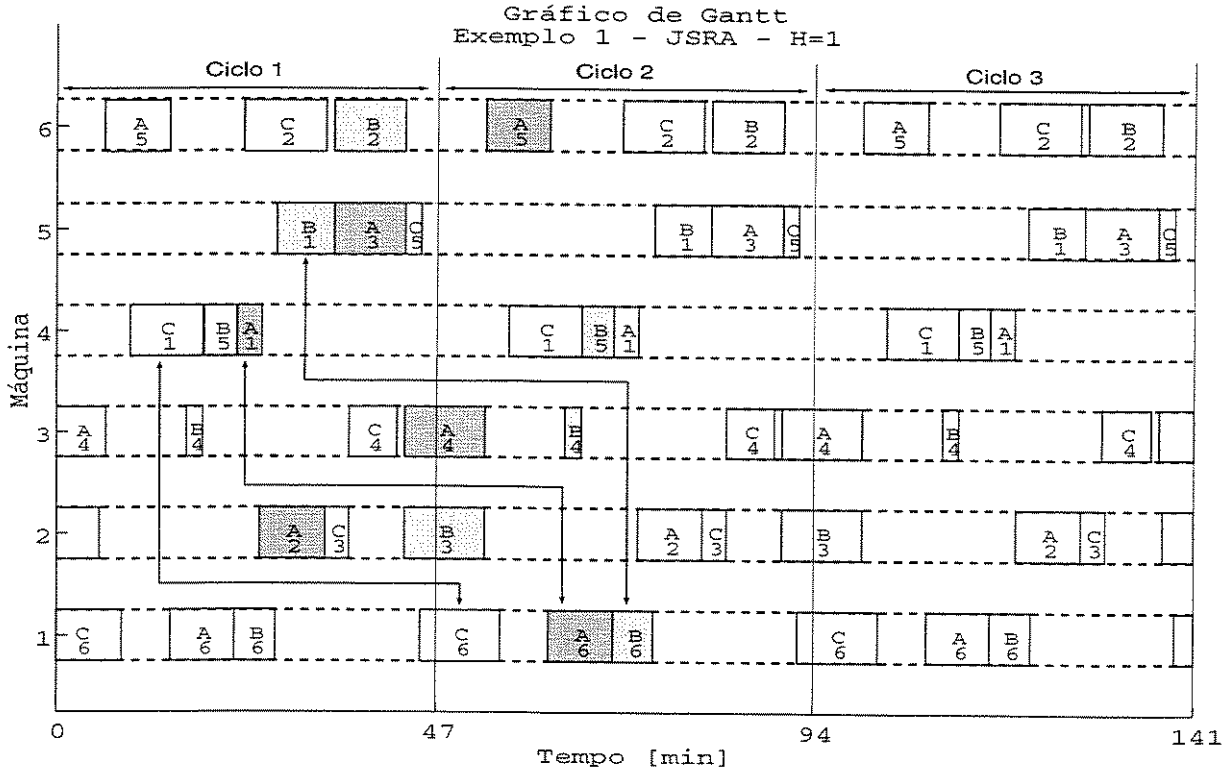
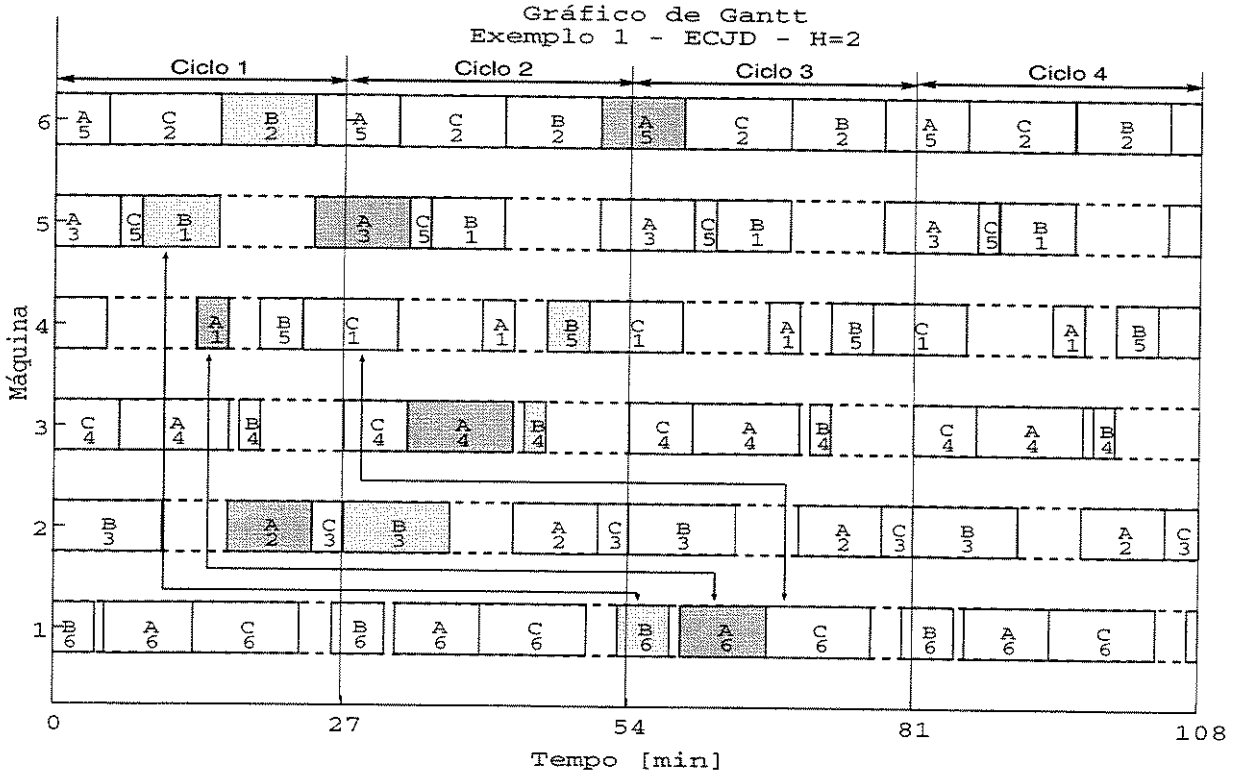




Figura 5.1: Escalonamento Ótimo do Exemplo 1 com h=1



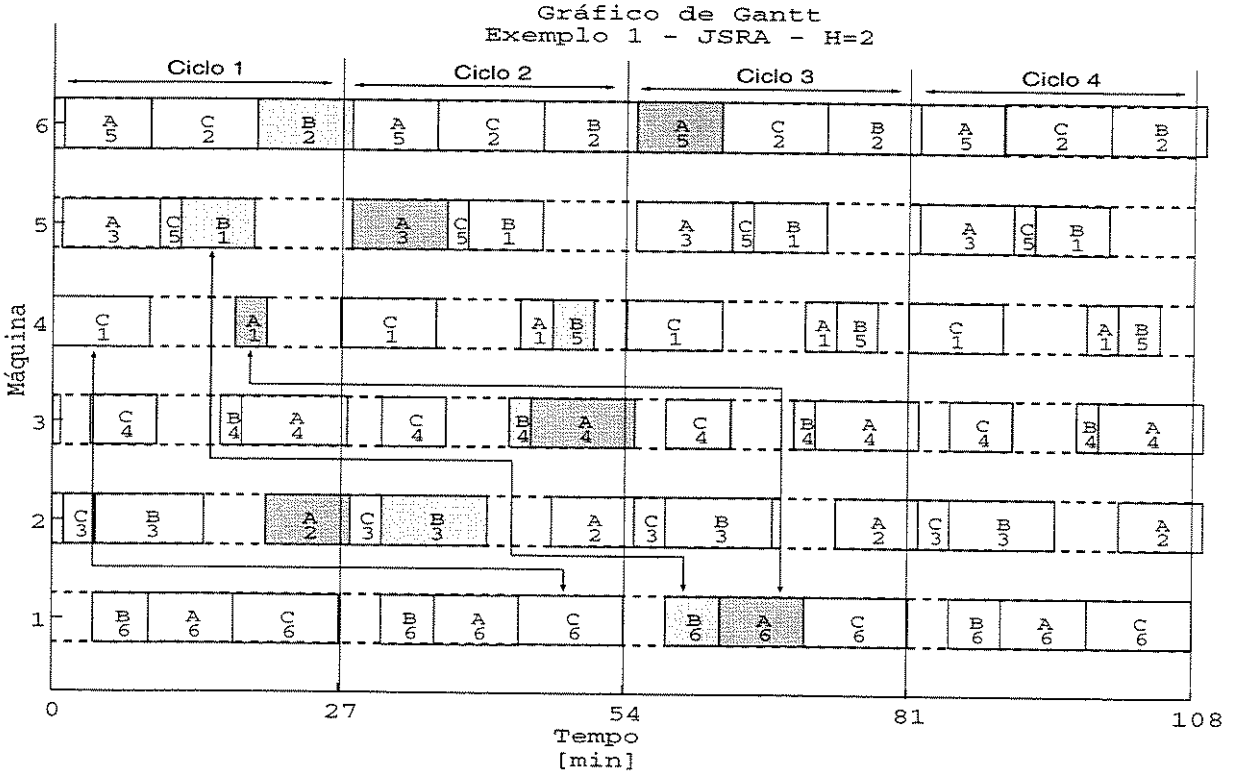




Figura 5.2: Escalonamento Ótimo do Exemplo 1 com h=2

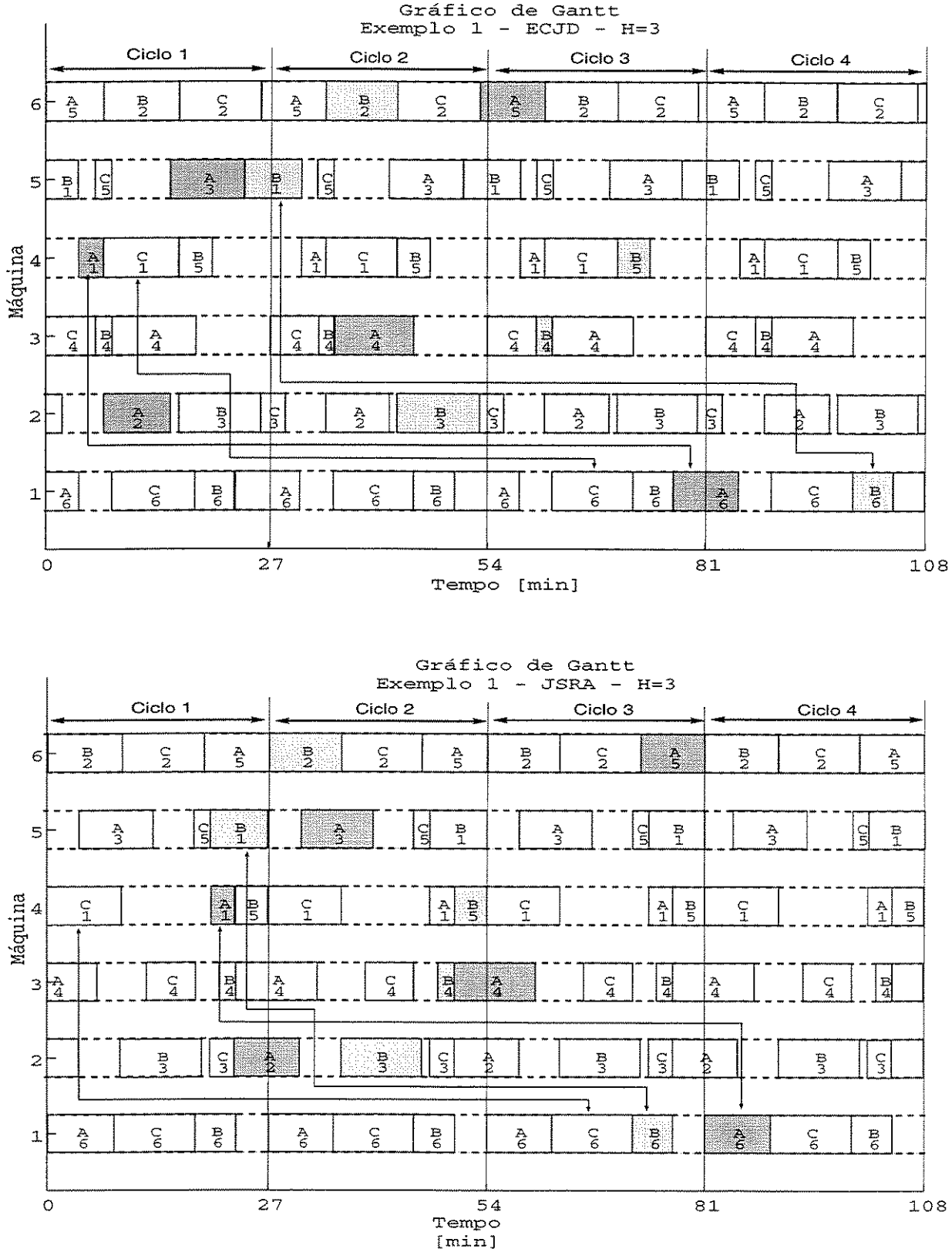



Figura 5.3: Escalonamento Ótimo do Exemplo 1 com h=3

Observando as cartas de Gantt vê-se que os dois modelos chegaram ao mesmo valor de ciclo ótimo, para cada caso avaliado, e embora, sendo diferentes, os escalonamentos ótimos são equivalentes. Pela figura 5.1 nota-se que a duração de qualquer *job* é igual ou inferior a Z=47, pela figura 5.2 a duração de qualquer *job* é igual ou inferior a 2Z=54, e finalmente pela figura 5.3 nota-se que a duração de qualquer *job* é igual ou inferior a 3Z=81. Mostrando dessa forma a influência da restrição de altura de recorrência.

No próximo exemplo será possível perceber melhor a relação entre *jobs* distintos, quando o comprimento de um *job* pode ser bem maior que o de outro *job*, observando o tempo em que cada instância de *jobs* distintos permanece no sistema.

**Exemplo 5.2** *: Serão produzidos três produtos, cada produto contendo, respectivamente, 5, 10 e 9 partes que devem ser executadas obedecendo uma certa ordem. Cada parte do produto é associada a uma única máquina e seu tempo de processamento é fixo. Pode-se associar: produto a job, e parte do produto a operações. Uma vez que uma operação começou a ser executada não pode ser interrompida para a execução de outra operação.*

Os dados do problema são mostrados na tabela a seguir [2] [3] :

| Job | Operação | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | O1 | O2 | O3 | O4 | O5 | O6 | O7 | O8 | O9 | O10 |
| J1 | (11,1) | (6,3) | (15,1) | (2,4) | (11,1) | (.) | (.) | (.) | (.) | (.) |
| J2 | (12,1) | (11,5) | (9,3) | (3,4) | (10,3) | (12,1) | (15,5) | (8,3) | (3,2) | (14,5) |
| J3 | (6,3) | (13,5) | (6,3) | (3,4) | (9,3) | (1,4) | (4,2) | (15,1) | (4,2) | (.) |

Tabela 5.2: Dados do exemplo 5.2

O objetivo é maximizar o fluxo do sistema, ou respectivamente, minimizar o período para a repetição das tarefas. Note que cada operação aparece apenas uma vez por período.

O problema será resolvido pelo programa **GAMS**, de duas maneiras, uma utilizando o modelo JSRA e a outra o ECJD. Os resultados serão mostrados em forma de carta de Gantt, geradas pelo software MatLab através dos arquivos de saída do **GAMS**. Nas cartas, as linhas verticais representam o ciclo ótimo, as linhas pontilhadas horizontais as máquinas e as setas mostram a duração de uma instância de cada *job* distinto.

[2] $(p_i,m_i)$ = (tempo de processamento da operação $i$, máquina associada a operação $i$)

[3] (.) = não existência da operação

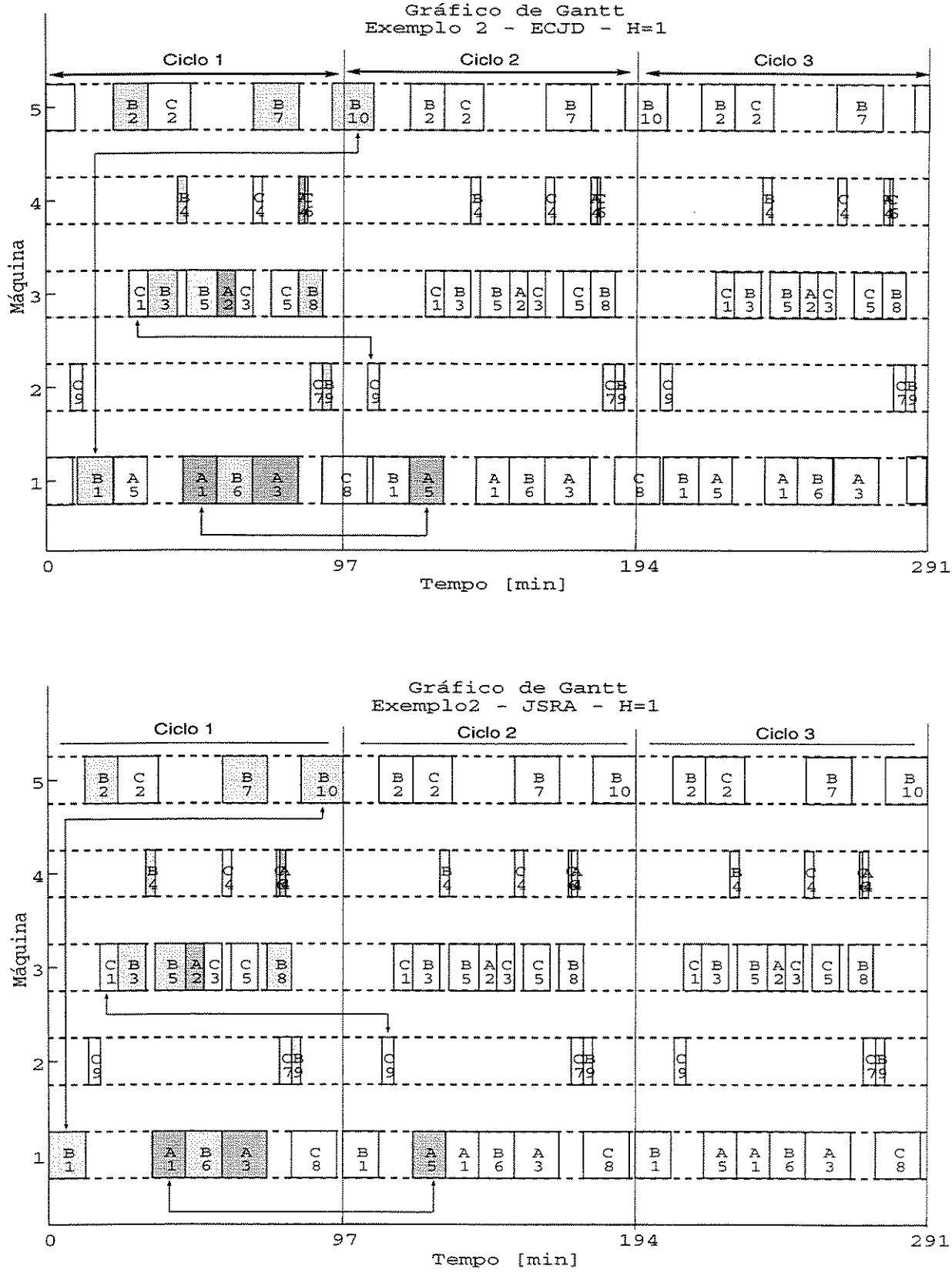
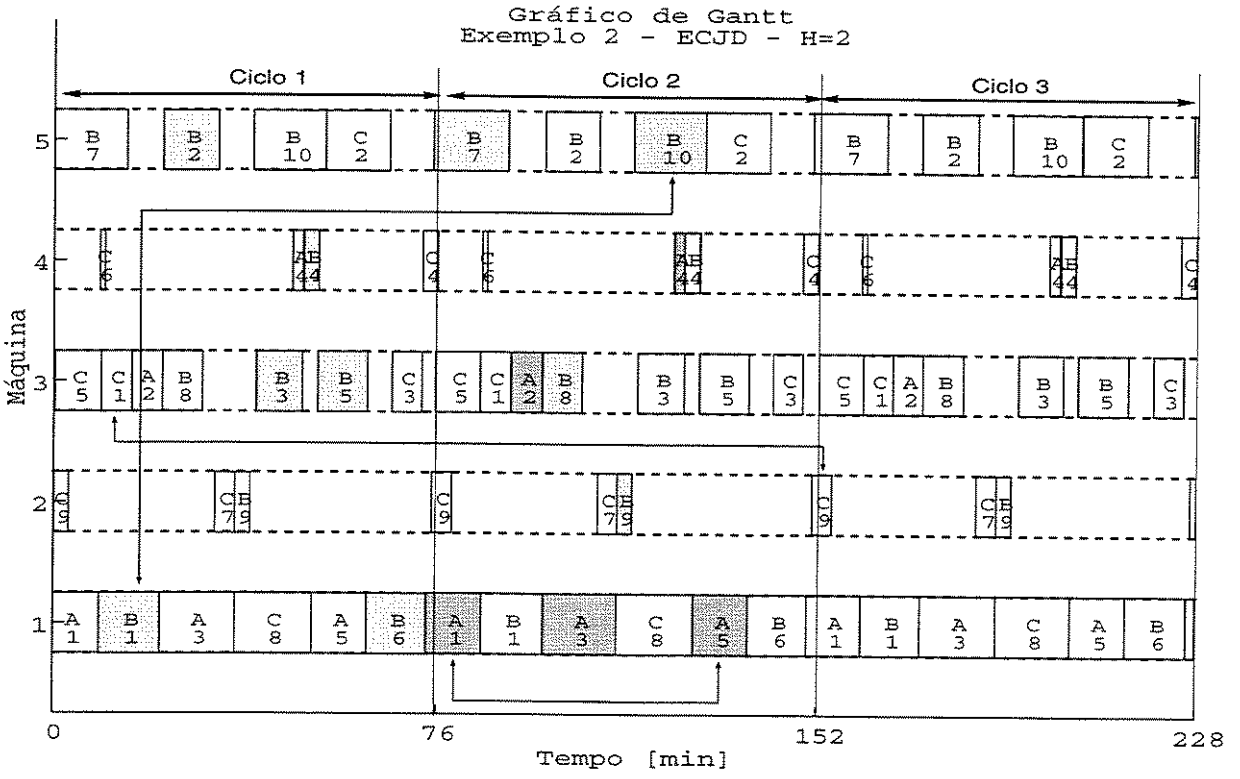



Figura 5.4: Escalonamento Ótimo do Exemplo 2 com h=1



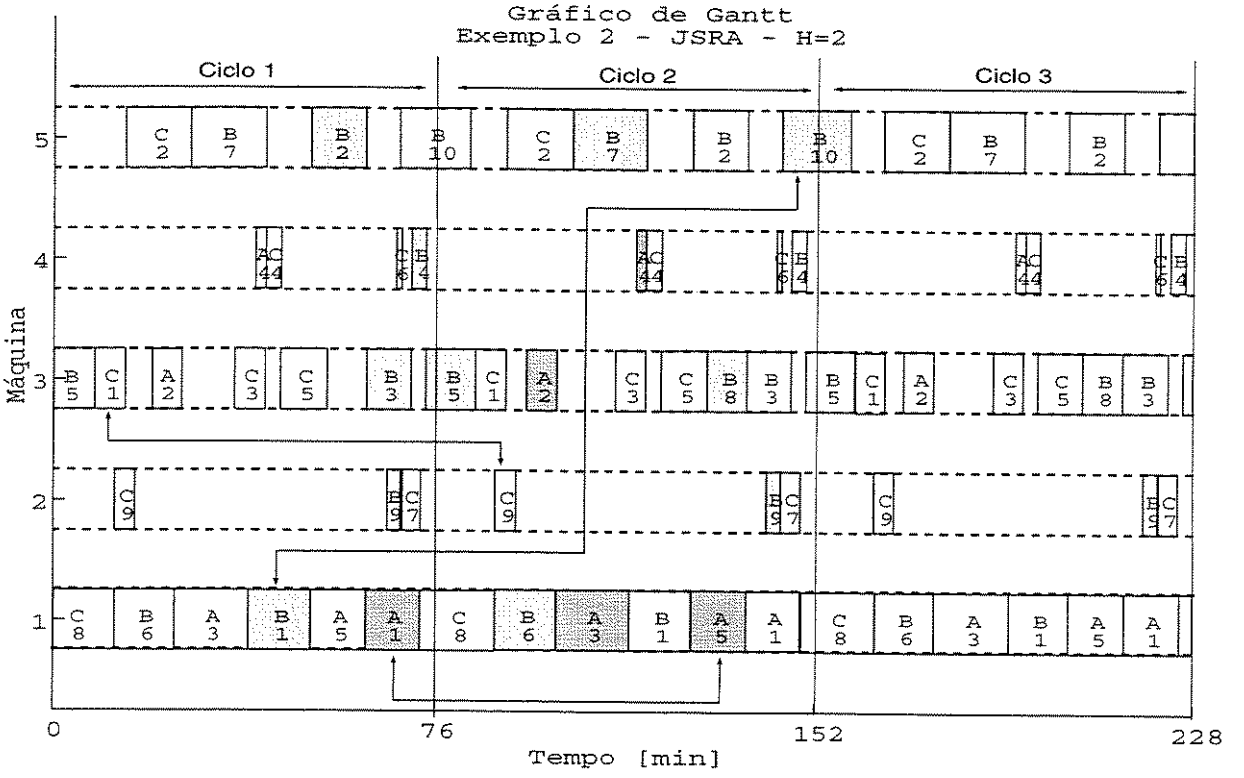




Figura 5.5: Escalonamento Ótimo do Exemplo 2 com h=2

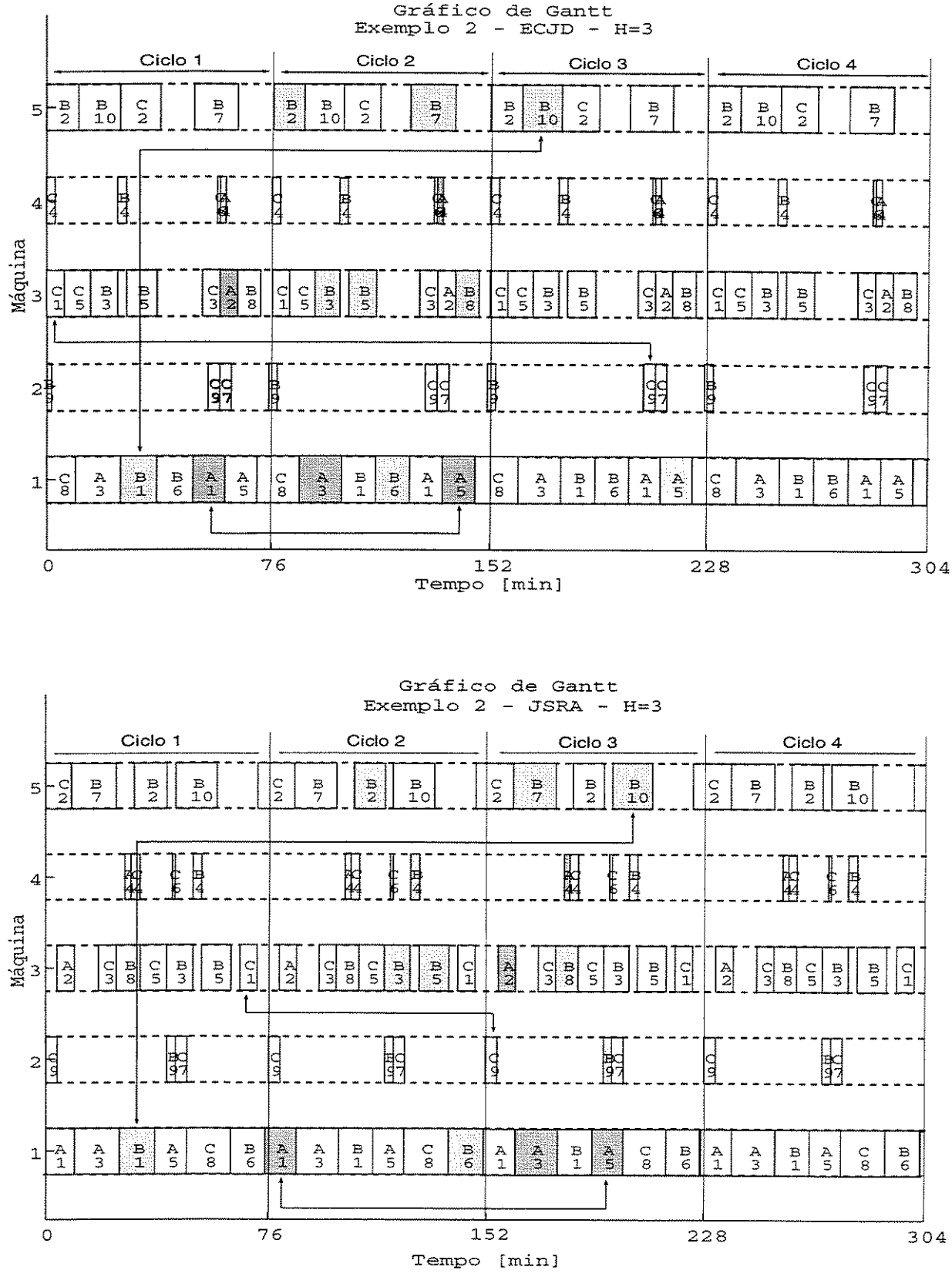



Figura 5.6: Escalonamento Ótimo do Exemplo 2 com h=3

Observando as cartas de Gantt vê-se que os dois modelos chegaram ao mesmo valor de ciclo ótimo, para cada caso avaliado, e embora, sendo diferentes, os escalonamentos ótimos são equivalentes. Pode-se notar ainda que como os *jobs* distintos tem comprimentos mínimos distintos, ao se restringir a altura de recorrência do *job* , equivalentemente controlando a largura, tende-se a aumentar o tempo do ciclo como pode ser verificado comparando-se o ciclo ótimo conseguido com h=1 e com h=2, que foram respectivamente,$Z_{h=1} = 97$ e $Z_{h=2} = 76$; porém quando se compara com o ciclo ótimo conseguido com h=2 e h=3, nota-se que o tempo de ciclo é igual, isto porque a máquina 1 já está sendo ocupada totalmente, isto é, não existe tempo de folga na seqüência entre as operações associadas a esta máquina.

Pode ainda ser verificado, em todas as figuras, que quando a "altura de recorrência real" de um *job* é igual a 1, a largura deste mesmo *job* também é igual a 1, quando a altura é 2 a largura é exatamente 2 e finalmente quando a altura é 3 sua largura também é 3 comprovando a equivalência dos atributos.

Estas três últimas figuras reforçam mais ainda a idéia da equivalência dos modelos ECJD e JSRA, e no próximo capítulo será mostrada analiticamente esta equivalência.

# 6

# Equivalência dos Modelos ECJD e JSRA

O modelo de escalonamento cíclico de *jobs* distintos (ECJD) é da seguinte forma:

$$\text{Minimizar } Z = f(T) \text{ sujeito a } T \in \Theta \subset \mathbb{Z}^n, \tag{6.1}$$

onde a função $f$ representa a função objetivo, $Z$ é o ciclo, o conjunto $\Theta$ é o conjunto factível, definido por um conjunto de igualdades e desigualdades, e a variável $T$ representa os tempos de início das operações.

O modelo de job shop recorrente adaptado (JSRA) é da seguinte forma:

$$\text{Minimizar } Z = f(t) \text{ sujeito a } t \in \Gamma \subset \mathbb{Z}^n, \tag{6.2}$$

onde a função $f$ representa a função objetivo, $Z$ é o ciclo, o conjunto $\Gamma$ é o conjunto factível, definido por um conjunto de igualdades e desigualdades, e a variável $t$ representa os tempos de início das operações.

O objetivo deste capítulo é demonstrar a equivalência entre os dois modelos. Para isto, inicialmente estabelece-se a correspondência entre as variáveis $t$ (do modelo JSRA) e $T$ (do modelo ECJD) e em seguida mostra-se que mediante esta transformação os conjuntos $\Theta$ e $\Gamma$ são iguais.

## 6.1 Transformações de variáveis

Nas definições das variáveis dadas na seção 3.1.3 nota-se que os modelos ECJD e JSRA definem a variável *tempo de inicialização da operação* ($T$ e $t$ respectivamente) de forma diferente. Para que os dois modelos sejam equivalentes é necessário estabelecer uma correspondência entre as variáveis. Para fazer isto é preciso utilizar-se o roteiro de fabricação estabelecido no problema.

Um análise dos modelos ECJD e JSRA permite concluir que

$$T = t \bmod Z \tag{6.3}$$

com $T \geq 0$ e $t \geq 0$.

Contudo, para facilitar as demonstrações a seguir é necessário explicitar a parte inteira da operação "mod" da equação 6.3. Portanto, de maneira mais detalhada, dado um escalonamento com comprimento de ciclo $z$, para transformar $T$ em $t$ pode-se fazer:

$$T_{Ji} = t_{Ji} - z \sum_{g=1, g \in \mathbb{O}_J}^{i-1} r_{Jg} - b; \tag{6.4}$$

$$T_{J1} = t_{J1} - b; \tag{6.5}$$

$$b = \min \{t_{J1} \mid J \in \mathbb{J}\}; \tag{6.6}$$

onde

$$r_{Ji} = \begin{cases} 0 & \text{se } s(Ji) \text{ inicializa-se no mesmo ciclo que } Ji; \\ 1 & \text{se } s(Ji) \text{ inicializa-se no ciclo seguinte relativo ao início da operação } Ji; \\ 2 & \text{caso contrário;} \end{cases}$$

e $s(Ji)$ é o sucessor de $Ji$ dado pelo roteiro de fabricação .

Para transformar $T$ em $t$ também é necessário recuperar a informação da parte inteira do quociente da divisão de $t$ por $Z$ de forma que $T + D * Z = t$. Para resolver isto será utilizada a variável $r_{Ji}$ descrita acima. Dado um escalonamento com comprimento de ciclo $z$, para transformar $T$ em $t$ pode-se fazer:

$$t_{Ji} = T_{Ji} + z \sum_{g=1, g \in \mathbb{O}_J}^{i-1} r_{Jg} \tag{6.7}$$

$$t_{J1} = T_{J1} \quad \forall J \in \mathbb{J} \tag{6.8}$$

## 6.2 Funcionalidade e classificação das restrições

Os conjuntos de restrições dos dois modelos utilizados para tratar o problema abordado neste trabalho podem ser particionados nos quatro conjuntos de restrições a seguir:

1. restrições de não sobreposição de operações em uma máquina;
2. restrições de fluxo, isto é, da altura de recorrência entre *jobs* distintos e entre instâncias de um mesmo *job*;
3. restrições de precedência das operações de um mesmo *job* e de limites, tanto superior como inferior das variáveis;

4. restrições física da máquina, isto é, de unicidade de predecessor e sucessor de cada operação na máquina, de não existência de subciclos e de existência de uma única última operação em cada máquina no ciclo.

O conjunto das soluções factíveis do problema é a intersecção dos 4 conjuntos acima.

As restrições associadas aos modelos JSRA e ECJD apresentadas nas seções 3.2 e 3.2 do capítulo 3 estão relacionadas com estes conjuntos da seguinte forma:

| Conjuntos de restrições | Equações dos Modelos | |
|---|---|---|
| | ECJD | JSRA |
| não sobreposição | 3.11 e 3.12 | 3.2, 3.3 e 3.4 |
| altura de recorrência | 3.16, 3.17, 3.18 e 3.19 | 3.1 e 3.5 |
| precedência e limites | 3.10, 3.20, 3.21, 3.22,3.23 e 3.25 | 3.1, 3.7 e 3.8 |
| físicas | 3.13, 3.14, 3.18 e 3.24 | implícitas |

Tabela 6.1: Conjuntos de restrições

Considerando que $\Theta$ é o conjunto de soluções factíveis de ECJD, Seção 3.2, e $\Gamma$ é o conjunto de soluções factíveis de JSRA, Seção 3.2, os seguintes conjuntos de soluções serão associados aos conjuntos de restrições descritos anteriormente:

| Conjunto de restrições | Conjunto Solução | |
|---|---|---|
| | ECJD | JSRA |
| não sobreposição | $\Theta_1$ | $\Gamma_1$ |
| altura de recorrência | $\Theta_2$ | $\Gamma_2$ |
| precedência e limites | $\Theta_4$ | $\Gamma_4$ |
| físicas | $\Theta_3$ | $\Gamma_3$ |
| | $\bigcap_1^4 i\, \Theta_i = \Theta$ | $\bigcap_1^4 i\, \Gamma_i = \Gamma$ |

Tabela 6.2: Conjuntos Solução

## 6.3 Teorema de Equivalência

Mostra-se nesta seção que os conjuntos $\Theta$ e $\Gamma$ são iguais, isto é, que toda solução factível em um dos problemas também é factível no outro problema. Em outras palavras:

$$\Theta = \Gamma.$$

Assim o seguinte teorema é proposto:

**Teorema 6.1** *(Equivalência): Os modelos de escalonamento cíclico de jobs distintos (ECJD) e de job-shop recorrente adaptado (JSRA) são equivalentes .*

**Demonstração:** Segundo o Lema B.2[1] o conjunto $\Theta_2$ é igual a $\Gamma_2$ mediante a transformação de variáveis definida na seção 6.1, isto é:

$$\Theta_2 = \Gamma_2.$$

E pelo Lema B.4[2] os conjuntos $\Theta_1 \cap \Theta_3 \cap \Theta_4$ e $\Gamma_1 \cap \Gamma_3 \cap \Gamma_4$ são iguais mediante a transformação de variáveis definida na seção 6.1, isto é:

$$\Theta_1 \cap \Theta_3 \cap \Theta_4 = \Gamma_1 \cap \Gamma_3 \cap \Gamma_4.$$

Logo é verdade que:

$$\Theta_1 \cap \Theta_2 \cap \Theta_3 \cap \Theta_4 = \Gamma_1 \cap \Gamma_2 \cap \Gamma_3 \cap \Gamma_4.$$

Portanto

$$\Theta = \Gamma$$

Em conclusão os conjuntos $\Theta$ e $\Gamma$ são iguais sob as transformações definidas nas seção 6.1 o que prova o teorema.

[1] Ver Apêndice B
[2] Ver Apêndice B

# 7

# Conclusões

O problema de escalonamento cíclico vem sendo cada vez mais estudado, porém, ainda poucos estudos sobre modelagem e formulação foram feitos. E por causa deste fato foram comparados, apenas, estes dois modelos. Neste estudo foram mostradas algumas características de problemas periódicos como a largura de um *job* e a altura de recorrência entre operações e através destas características conseguiu-se a equivalência entre os modelos ECJD e JSRA.

A equivalência entre os modelos foi mostrada, tanto analítica como empiricamente. Apesar disso, na resolução dos experimentos, os modelos não encontraram os mesmos escalonamentos cíclicos, mas sim, escalonamentos equivalentes com o mesmo período ótimo. Outra diferença notada foi a quantidade de iterações necessárias para se encontrar a solução ótima, que no caso do JSRA foi, em grande parte, sempre superior ao ECJD. Porém o modelo JSRA conseguiu encontrar sempre uma solução factível em poucas iterações, enquanto o ECJD dependeu bastante dos valores de B e n adotados.

Com isso, algumas idéias surgiram, como por exemplo, usar as formulações do ECJD e JSRA para modelar problemas de grande porte, como o de despacho de trens de minério e cargueiros em uma linha singela, explorando a variação do fluxo através da altura de recorrência.

Aqui trabalhou-se sempre com máquinas distintas, isto é, cada operação só poderia ser executada por uma única máquina; mas poder-se-ia explorar a relação dos modelos num ambiente com máquinas idênticas, onde uma operação seria associada a um único conjunto de máquinas. Esta relação poderia ser feita tendo em vista problemas em processamento paralelo.

# Referências Bibliográficas


Audsley, N., K. Tindell, and A. Burns(1993). The End of The Line for Static Cyclic Scheduling? *Proceedings of 5th Euromicro Workshop on Real-Time Systems*, Oulu, Finland, 36–41.

Audsley, N., & A. Burns (1990). Real-Time System Scheduling. Report YCS 134, Department of Computer Science, University of York, UK.

Baker K.R. (1974). *Introduction to Sequencing and Scheduling.* John Wiley & Sons.

Bar-Noy, A., R. Bhatia, J.(S.) Naor and B. Schieber (1998). Minimizing Service and Operation Costs of Periodic Scheduling. *Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA).* San Francisco, California, USA.

Bartholdi III, J.J., J.B. Orlin, and H.D. Ratliff (1980). Cyclic Scheduling via integer Programs with Circular Ones. *Operations Research*, Vol. 28, Nº5, 1074–1085.

Bartholdi III, J.J. (1981). A Guaranteed-Accuracy Round-Off Algorithm for Cyclic Scheduling and Set Covering. *Operations Research*, Vol. 29, Nº3, 501–510.

Baruah, S.K., J.E. Gehrke and C.G. Plaxton (1995). Fast Scheduling on Periodic Tasks on Multiple Resources. Technical Report CS-TR-95-02, University of Texas, Austin, USA.

Burkard, R.E. (1986). Optimal Schedules for Periodically Recurring. *Discrete Applied Mathematics*, **15**, 167–180.

Calland, P.Y., A. Darte and Y. Robert (1995). A New Guaranteed Heuristic for the Software Pipelining Problem. Rapport de Recherche 2759, INRIA Rhone-Alpes, Grenoble, France.

Carter, M.W., M.J. Magazine and T-S. Moon (1988). Technical Note: A Warning about Cyclic Lot Scheduling with Sequence Dependent Set-Up Times. *Int. J. Prod. Res.*, Vol. 26, Nº7, 1281–1283.

Changyou, L. (1990). Modelling and Analizing a Class of Flexible Manufacturing System. *Lecture Notes in Control and Information Sciences*, **144**, 985–992.

Chretienne, P. (1991). The Basic Cyclic Scheduling Problem with Deadlines. *Discrete Applied Mathematics*, **30**, 109–123.

Conway, R.W., W.L. Maxwell, L.W. Miller (1967). *Theory of Scheduling.* Addison-Wesley, Reading, Mass.

Crama, Y. & J.V. Klundert (1997). Cyclic Scheduling of Identical Parts in a Robotic Cell. *Operations Research*, Vol. 45, Nº6, 952–965.

Dantzig, G., D. Fulkerson and S. Johnson (1954). Solution of a large scale traveling salesman problem. *Operations Research*, Vol. 2, Nº6, 393–410.

Dauscha, W., H.D. Modrow and A. Neumann (1985). On Cyclic Sequence Types for Constructing Cyclic Schedules. *Zeitschrift für Operations Research*, **29**, 1–30.

Dobson, G.(1992). The Cyclic Lot Scheduling Problem with Sequence-Dependent Setups. *Operations Research*, Vol. 40, Nº4, 736–749.

Emmons, H.(1985). Work-Force Scheduling with Cyclic Requirements and Constraints on Days off, Weekends Off, Work Stretch. *IIE Transactions*, Vol. 17, Nº1, 8–16.

Gasperoni, F. & U. Schwiegelshohn (1991). Efficient Algorithms for Cyclic Scheduling . Technical Report 1991-571, Courant Institute of Mathematical Sciences, New York.

Gaujal, B. (1994). Regular Sequences and Applications in Cyclic Scheduling . *Proceedings of the 4th QMIPS Workshop on the Performance of Parallel Computer Systems*, London, England, 25–48.

Gondhalekar, V.A. (1995). Scheduling Periodic Wireless Data Broadcast. Technical Report CS-TR-95-43, University of Texas, Austin, USA.

Grigor'Yeva, N.S., I.SH. Latypov and I.V. Romanovskiy (1989). Cyclic Problems of Scheduling Theory. *Soviet Journal of Computer and Systems Sciences*, Vol. 27, Nº2, 34–39.

Groth, A.F. & R.S. Mendes (1997). A Min-Max Model for Reactive Job Shop Scheduling. *4th European Control Conference*, Bruxelas

Hall, R.W. (1988). Scheduling for Improvement. *Int. J. Prod. Res.*, Vol. 26, Nº3, 457–472.

Hanen, C. (1989). Optimizing Microprograms for Recurrent Loops on Pipelined Architectures Using Timed Petri Nets. In G. Rozenberg (ed.), *Advances in Petri Nets*, Springer–Verlag, Berlin, 236–261.

Hanen, C. (1994). Study of a NP-hard Cyclic Scheduling Problem: The Recurrent Job-Shop. *Eur. J. Oper. Res.*, **72**, 82–101.

Hanen, C. & S. Munier (1992). A study of The Cyclic Scheduling Problem on Parallel Processors. Rapport LRI 766, Universitè Paris–sud, Paris.

Hanen, C. & S. Munier (1993). Cyclic Scheduling on Parallel Processors: An Overview. LRI Internal Report, Universitè Paris–sud, Paris.

Kamoun, H. & C. Sriskandarajah (1993). The Complexity of Scheduling Jobs in Repetitive Manufacturing Systems. *Eur. J. Oper. Res.*, **70**, 35–364.

Karabati, S., P. Kouvelis and A.S. Kiran (1992). Games, Critical Paths and Assignment Problems om Permutation Flow Shop and Cyclic Scheduling Flow Line Environments. *J. Opl. Res. Soc.*, Vol. 43, nº3, 241–258.

Karabati, S. & P. Kouvelis (1996). Cyclic Scheduling in Flow Lines: Modelling Observations, Effective Heuristics amd a Cycle Time Minimization Procedure. *Naval Research Logistics*, **43**, 211–231.

Katz, V., & E. Levner (1997). Minimizing the Number of Robots to Meet a Given Cyclic Schedule.*Annals of Operations Research*, **69**, 227–239.

Katz, V., & E. Levner (1997a). A Strongly Polynomial Algorithm for No-Wait Cyclic Robotic Flowshop Scheduling . *Operations Research Letters*, **21**, 171–179.

Korst, J., E. Aarts, J.K. Lenstra and J. Wessels (1991). Periodic Multiprocessor Scheduling . *Lecture Notes in Computer Science*, **505**, 166–178.

Kouvelis, P. & S. Karabati (1997). Cyclic Scheduling in Synchronous Production Lines. Working Paper 1997-16, Koç University, College of Administrative Sciences and Economics, Instinye, Istambul,Turkey.

Lawler,E.L., J.K.Lenstra, A.H.G.Rinnooy Kan and D.B.Shmoys (1985). *The Traveling Salesman Problem, A Guided Tour of Combinatorial Optimization*, Wiley.

Lee, T–E. & M.E. Posner (1997). Performance Measures and Schedules in Periodic Job Shops. *Operations Research*, Vol. 45, Nº1, 72–91.

Lee, T–E. (1994). Stable Earliest Schedules for Periodic Job Shops: A Linear System Approach. In *Proceedings of the 11th International Conference on Analysis and Optimization of Systems. Lecture Notes in Control and Information Science*, **199**, Springer, Berlin, 290–296.

Lei, L. & T–J. Wang (1991). The Minimum Common-Cycle Algorithm for Cyclic Scheduling of Two Material Handling Hoists with Time Window Constrainst. *Management Science*, Vol. 37, Nº12, 1629–1639.

Lei, L., R. Armstrong and S. Gu(1993). Minimizing the Fleet Size with Dependent Time-Window and Single-Track Constraints. *Operations Research Letters*, Vol. 14, Nº14, 91–98.

Levner, E. & V. Kats (1998). A Parametric Critical Path Problem and an Aplication for Cyclic Scheduling. *Discrete Applied Mathematics*, **87**, 149–158.

Loerch, A.G. & J.A. Muckstadt (1994). An Approach to Production Planning and Scheduling in Cyclically Scheduled Manufacturing Systems. *Int. J. Prod. Res.*, Vol. 32, Nº4, 851–871.

Maxwell, W.L., J.A. Muckstadt, P.L. Jackson and R.O. Roundy (1986). The Interaction Between Design and Scheduling in Repetitive Manufaturing Environments. Technical Report 692, SORIE, Cornell University, Ithaca, New York, USA.

Miller, C.E., A.W. Tucker and R.A. Zemlin (1960). Integer Programming Formulation of Traveling Salesman Problem. *J. ACM.*, Vol. 7, Nº4, 326–329.

Melamed, I.I., S.I. Sergeev and I.KH. Sigal (1989). The Traveling Salesman Problem. 1. Theoretical Issues. *Automation and Remote Control*, Vol. 50, Nº9, 1147–1173.

Munier, A. (1996a). The Complexity of a Cyclic Scheduling Problem with Identical Machines and Precedence Constraints. *European Journal of Operations Research*, **91**, 471-480.

Munier, A. (1996b). The Basic Cyclic Scheduling Problem with Linear Precedence Constraints. *Discrete Applied Mathematics*, **64**, 219-238.

Munshi, A.A. & B. Simons (1990). Scheduling Sequential Loops on Parallel Processors. *SIAM J. Comput.*, Vol. 19, Nº4, 728–741.

Nawijn, W.M. (1998). A Cyclic Production Problem: An Application of Max-Algebra. *Eur. J. Oper. Res.*, **104**, 187–200.

Nori, V.S. & B.R. Sarker(1996). Cyclic Scheduling for a Multi-Product, Single-Facility Production System Operating Under a Just-in-Time Delivery Policy.*J. Oper. Res. Soc.*, **47**, 930–935.

Orlin, J.B. (1982). Minimizing the Number of Vehicles to Meet a Fixed Periodic Schedule: An Application of Periodic Posets. *Operations Research*, Vol. 30, Nº4, 760–776.

Park, K.S. & D.K. Yun (1985). Optimal Scheduling of Periodic Activities. *Operations Research*, Vol. 33, Nº3, 690–695.

Pinto, J.M. & I.E. Grossmann (1994). Optimal Cyclic Scheduling of Multistage Continuous Multiproduct Plants. *Computers Chem. Engng.*, Vol. 18, Nº9, 797–816.

Rao, U.S. (1992). Multi-Stage, Identical Job Cyclic Scheduling for Repetitive Manufacturing. Technical Report 1029, SORIE, Cornell University, Ithaca, NY, USA.

Rao, U.S. & P.L Jackson (1991). Subproblems in Identical Job Cyclic Scheduling, Properties, Complexity and Solution Approaches. Technical Report 956, SORIE, Cornell University, Ithaca, NY, USA.

Rao, U.S. (1993). A Study of Sequencing Heuristics for Periodic Production Environments. Technical Report 1077, SORIE, Cornell University, Ithaca, NY, USA.

Risset, T., F.D. Dinechin and S. Robert (1997). Structured Scheduling of Recurrence Equations. Rapport de Recherche 3282, IRISA, Beaulieu, INRIA Rennes, France.

Roundy, R. (1987). Cyclic Schedules for Job Shops with Identical Jobs. Technical Report 766, SORIE, Cornell University, Ithaca, NY, USA.

Roundy, R. (1992). Cyclic Schedules for Job Shops with Identical Jobs. *Mathematics of Operations Research*, Vol. 17, Nº4, 842–865.

Sahinidis, N.V. & I.E. Grossmann (1991). MINLP Model for Cyclic Multiproduct Scheduling on Continuous Parallel Lines. *Computers Chem. Engng.*, Vol. 15, Nº2, 85–103.

Serafini, P. & W. Ukovich (1989). A Mathematical Model for Periodic Scheduling Problems. *SIAM J. Disc. Math.*, Vol. 2. Nº4, 550–581.

Silver, E.A. (1995). Dealing with a Shelf Life Constraint in Cyclic Scheduling by Adjusting both Cycle Time and Production Rate. *Int. J. Prod. Res.*, Vol. 33, Nº3, 623–629.

Song, J.S. & T.E. Lee (1996). A Tabu Search Procedure for Periodic Job Shop Scheduling. *Computers Ind. Engng*, Vol. 30, Nº3, 433–447.

Song, J.S. & T.E. Lee (1998). Petri Net Modeling and Scheduling for Cyclic Job Shops with Blocking. *Computers Ind. Engng*, Vol. 34, Nº2, 281–295.

Tayur, S. (1996). Cyclic Schedules as Part of a JIT Implementation at a Laminate Plant. Working Paper 1996–E37, GSIA, Carnegie Mellon University, Pittsburgh, USA.

Verhaegh, W.F.J., P.E.R. Lippens, E.H.L. Aarts, J.L. van Meerbergen and A. van der Werf (1998). The Complexity of Multidimensional periodic Scheduling. *Discrete Applied Mathematics*, **89**, 213–242.

Vince, A. (1989). Scheduling Periodic Events.*Discrete Applied Mathematics*, **25**, 299–310.

Yura, K. (1989). Multi-Job Cyclic Scheduling for an Automated Job-Shop-Type Manufacturing System (Relationship Between Job Output Interval and Buffer Capacity).*JSME International Journal*, Series III, Vol. 32, Nº3, 498–503.

Zhang, H. & S.C. Graves (1997). Cyclic Scheduling in a Sthochastic Environment. *Operations Research*, Vol. 45, Nº6, 894–903.

# Apêndices

# A

# Eliminação de Subciclos

Rao (1992) usa em sua formulação a seguinte restrição para eliminação de subconjunto cíclico em conjunto com as restrições de unicidade de predecessor e sucessor:

$$\sum_{(i,j)\in\mathbb{O}_{s'}} x_{ij} \leq |\mathbb{O}_{s'}| - 1 \quad \forall \mathbb{O}_{s'}, \quad \mathbb{O}_{s'} \subset \mathbb{O}_s, \mathbb{O}_{s'} \neq \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{A.1}$$

Esta equação foi proposta por Dantzig-Fulkerson-Johnson (Dantzig *et al.*, 1954) e gera $2^n - 2$ restrições, $n$ representando a quantidade de operações em uma máquina. A equação proposta por Miller *et al.*(1960) gera $(n-1)^2$ restrições como foi mostrado por Lawler *et al.*(1985):

$$v_i - v_j + nx_{ij} \leq (n-1) \quad \forall i, j \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}. \tag{A.2}$$

Mostra-se a seguir que é possível usar a equação A.2 (MTZ modificada) no lugar da equação A.1 (DFJ) no modelo proposto por Rao (1992).

Miller *et al.* (1960) propuseram o seguinte conjunto de restrições que garante a eliminação de *sub-tours* em um problema de caixeiro viajante:

$$v_i - v_j + nx_{ij} \leq (n-1) \quad \forall i, j \in \mathbb{C}, \tag{A.3}$$

$$\sum_{i=0}^{n} x_{ij} = 1 \quad j = 1 \ldots n, \tag{A.4}$$

$$\sum_{j=0}^{n} x_{ij} = 1 \quad j = 1 \ldots n, \tag{A.5}$$

$$\sum_{i=1}^{n} x_{i0} = t, \tag{A.6}$$

onde

$t$ – quantidade de retornos a cidade 0;

$n$ – total de cidades;

$\mathbb{C}$ – conjunto de cidades pertencentes a um *tour*;

Porém a equação A.2 não deixa que o ciclo se feche assim é preciso modificá-la para se adequar ao problema tratado aqui. Uma modificação possível e que gera $n^2 - n$ restrições é:

$$v_i - v_j + nx_{ij} \leq (n-1) + n(L_i + L_j) \quad \forall i, j \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}.$$

onde

$$L_{Ji} = \begin{cases} 1 & \text{se } Ji \text{ é a última operação do ciclo da máquina ;} \\ 0 & \text{caso contrário.} \end{cases}$$

E constrói-se a seguinte formulação baseada no modelo de Rao (1992) e denominada Miller-Tucker-Zemlin modificada:

$$v_i - v_j + nx_{ij} \leq (n-1) + n(L_i + L_j) \quad \forall i, j \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{A.7}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall i, j \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{A.8}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad \forall i, j \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{A.9}$$

$$\sum_{i=1}^{n} L_i = 1 \quad \forall i \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{A.10}$$

A equação (A.10) representa a restrição de unicidade da última operação $L_i$ de cada máquina. As restrições (A.8) e (A.9) representam a unicidade do predecessor e do sucessor de cada operação. Dessa forma fazendo uma associação com o modelo proposto em Miller *et al.* (1960) a cidade de origem será a última operação $L_i$ de cada máquina. A principal diferença é que $L_i$ não é conhecida a priori. As variáveis $v_{i's}$ da equação (A.7) criam uma regra similar aos dos nós potenciais (Miller *et al.*,(1960)) em uma rede e as desigualdades envolvendo-as servem para eliminar subconjuntos cíclicos.

**Lema A.1** : *Se uma seqüência satisfaz as restrições de Miller-Tucker-Zemlin modificada, então a seqüência é cíclica e não contém sub-ciclos.*

A seguinte diferenciação deve ser notada:

- Seqüência cíclica, refere-se a uma ordem de execução de operações, que pode ser dada ou calculada pelo problema, e que é periódica.
- Escalonamento cíclico, refere-se a uma ordem de início de execução de cada operação pertencente a uma seqüência cíclica.

**Demonstração:**

Toda seqüência que satisfaz as restrições de unicidade de predecessor e sucessor é cíclica. Para demonstrar isto, suponha, por absurdo, que exista uma seqüência que satisfaz as equações (A.8) e (A.9), porém não é cíclica. Então a primeira operação desta seqüência não possui predecessor e a última operação não possui sucessor o que contradiz as restrições de unicidade de predecessor e sucessor. Portanto toda seqüência satisfazendo estas restrições é cíclica.

Dada uma seqüência cíclica onde, por conveniência, $L_i$ é a operação inicial ($r_0 = i$ quando $L_i = 1$) tem-se:

$$r_0 \rightarrow r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_j \rightarrow r_{k-1} \rightarrow r_0.$$

Assim,

$$x_{r_0\, r_1} = x_{r_1\, r_2} = x_{r_2\, r_3} = \ldots = x_{r_j\, r_{k-1}} = x_{r_{k-1}\, r_0} = 1. \tag{A.11}$$

O ciclo acima deve satisfazer a restrição :

$$v_i - v_j + n x_{ij} \leq (n-1) + c(L_i + L_j) \quad \forall i, j \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{A.12}$$

onde c é uma constante.

A seqüência de operações dada pela equação (A.11) irão satisfazer a seguinte equação derivada da equação A.12:

$$v_i - v_j \leq -1 + c(L_i + L_j) \quad \forall i, j \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{A.13}$$

Somando todas as equações (A.13) geradas pela seqüência cíclica encontra-se $c \geq k/2$, isto é:

$$\begin{array}{rcl}
v_{r_0} - v_{r_1} & \leq & -1 + c \\
v_{r_1} - v_{r_2} & \leq & -1 \\
v_{r_2} - v_{r_3} & \leq & -1 \\
\vdots & & \\
v_{r_j} - v_{r_{k-1}} & \leq & -1 \\
v_{r_{k-1}} - v_{r_0} & \leq & -1 + c \\
\hline
v_{r_0} - v_{r_0} & \leq & -k + 2c \\
& \Rightarrow & c \geq k/2
\end{array} \tag{A.14}$$

logo qualquer $c \geq k/2$ torna o ciclo factível. Assim faz-se $c = n$.

Somando todas as equações (A.13) geradas por uma sequência cíclica que não contem $r_0$ chega-se a:

$$\begin{aligned} v_{r_1} - v_{r_2} &\leq -1 \\ v_{r_2} - v_{r_3} &\leq -1 \\ &\vdots \\ v_{r_j} - v_{r_k} &\leq -1 \\ v_{r_k} - v_{r_1} &\leq -1 \\ \hline v_{r_1} - v_{r_1} = 0 &\leq -k \end{aligned} \tag{A.15}$$

Como é impossível encontrar-se um número negativo maior que zero conclui-se que qualquer sequência cíclica factível contém a última operação $L_i$.

Agora suponha, por absurdo, que exista uma seqüência cíclica com k operações sendo $k < n$. Sabe-se pelo resultado anterior que a seqüência só será factível se contiver $r_0$. Dessa forma, existem $(n-k)$ operações que não participam desta seqüência mas que devem satisfazer as restrições A.8 e A.9 de sucessão e predecessão, assim tem-se um conjunto de $(n-k)$ operações que geram, no mínimo, uma seqüência que não contém a última operação $r_0$. Como é impossível encontrar-se tal seqüência conclui-se que toda seqüência cíclica contém as $n$ operações associadas à máquina.

Logo toda seqüência que satisfaz Miller-Tucker-Zemlin modificada é cíclica e contém as $n$ operações associadas à máquina.

**Lema A.2** : *Uma seqüência cíclica que satisfaz as equações de unicidade de predecessor e de sucessor, satisfaz as restrições de Miller-Tucker-Zemlin (MTZ) modificada se e somente se satisfaz as restrições de Dantzig-Fulkerson-Johnson (DFJ).*

**Demonstração:** Seja $s'$ uma seqüência cíclica que não satisfaz a restrição de DFJ e $\mathbb{O}_{s'}$ o conjunto de operações pertencentes a $s'$. Então:

$$\sum_{(i,j)\in\mathbb{O}_{s'}} x_{ij} > |\mathbb{O}_{s'}| - 1, \quad \mathbb{O}_{s'} \subset \mathbb{O}_s, \mathbb{O}_{s'} \neq \mathbb{O}_s, \quad \forall s \in \mathbb{M},$$

e como cada elemento de $s'$ possui somente um predecessor e um sucessor, tem-se:

$$\sum_{(i,j)\in\mathbb{O}_{s'}} x_{ij} = |\mathbb{O}_{s'}| \quad \mathbb{O}_{s'} \subset \mathbb{O}_s, \mathbb{O}_{s'} \neq \mathbb{O}_s, \quad \forall s \in \mathbb{M}.$$

Como $\mathbb{O}_{s'}$ forma uma subseqüência cíclica de $\mathbb{O}_s$ e pelas equações A.8 e A.9 (de unicidade de predecessor e sucessor respectivamente) pode-se concluir que deve existir pelo menos uma subseqüência cíclica $\mathbb{O}_{s''}$ que está em $\mathbb{O}_s$, mas é diferente de $\mathbb{O}_{s'}$, isto é:

$$\mathbb{O}_{s''} = \mathbb{O}_s - \mathbb{O}_{s'}.$$

As seqüências $s'$ e $s''$ devem satisfazer respectivamente:

$$v_i - v_j + nx_{ij} \leq (n-1) + n(L_i + L_j) \quad \forall i,j \in \mathbb{O}_{s'}, \mathbb{O}_{s'} \subset \mathbb{O}_s, \mathbb{O}_{s'} \neq \mathbb{O}_s, s \in \mathbb{M}, \tag{A.16}$$

$$v_i - v_j + nx_{ij} \leq (n-1) + n(L_i + L_j) \quad \forall i,j \in \mathbb{O}_{s''}, \mathbb{O}_{s''} \subset \mathbb{O}_s, \mathbb{O}_{s''} \neq \mathbb{O}_s, s \in \mathbb{M}. \tag{A.17}$$

Porém pelos resultados dados pelas equações (A.14) e (A.15) na prova do Lema A.1 as equações (A.16) e (A.17) só serão satisfeitas se contiverem a ultima operação $L_i$ e como $L_i$ não pode pertencer as duas subseqüências cíclicas ao mesmo tempo, então não é possível encontrar-se $s'$ e $s''$ de forma a satisfazer as restrições de MTZ modificada.

Logo se uma seqüência cíclica $s'$ não satisfaz DFJ então não satisfaz as restrições de MTZ modificada.

Seja $s'$ uma seqüência cíclica que não satisfaz as restrições de MTZ modificada (A.7) e (A.10), mas que satisfaz as equações (A.8) e (A.9) e seja $\mathbb{O}_{s'}$ o conjunto de operações pertencentes a $s'$. Pelos resultados (A.14) e (A.15) na prova do Lema A.1 sabe-se que apenas se a seqüência cíclica $s'$ não contiver $L_i$ não satisfará MTZ modificada, assim se $s'$ não contém $L_i$ então $\mathbb{O}s' \subset \mathbb{O}s$, dessa forma $s'$ é uma subseqüência cíclica e:

$$\sum_{(i,j)\in\mathbb{O}_{s'}} x_{ij} = |\mathbb{O}_{s'}| > |\mathbb{O}_{s'}| - 1 \quad \mathbb{O}_{s'} \subset \mathbb{O}_s, \mathbb{O}_{s'} \neq \mathbb{O}_s, \quad \forall s \in \mathbb{M}.$$

Logo $s'$ não satisfaz DFJ.

Portanto se uma seqüência cíclica $s'$ não satisfaz MTZ modificada então não satisfaz as restrições de DFJ.

Pelo Lema A.1 e pelo Lema A.2 conclui-se que toda seqüência cíclica factível em DFJ satisfaz MTZ modificada e toda seqüência cíclica factível em MTZ modificada satisfaz DFJ. Dessa forma pode-se substituir DFJ por MTZ modificada no modelo de Rao (1992).

# B

# Equivalências

Os conjuntos utilizados neste apêndice são os mesmos conjuntos de restrições definidos no Cap. 6 e as definições das notações e das variáveis são dadas no Cap.3. A seguir serão feitas as demonstrações da validade dos lemas utilizados no Cap. 6.

**Lema B.1** : *Os conjuntos* $\Theta_1$ *e* $\Gamma_1$ *são iguais.*

**Demonstração:** No modelo JSRA as equações que representam as restrições de não sobreposição de operações na máquina e que correspondem as da seção 6.2 são:

$$t_{Qj} - t_{Ji} \geq p_{Ji} - k(Ji, Qj)z \quad \forall Ji, j \in \mathbb{O}_m, \quad \forall m \in \mathbb{M}, \tag{B.1}$$

$$k(Ji, Qj) + k(Qj, Ji) = 1 \quad \forall Ji, Qj \in \mathbb{O}_m, \quad \forall m \in \mathbb{M}, \tag{B.2}$$

$$k(Ji, Qj) \in \mathbb{Z} \quad \forall Ji, Qj \in \mathbb{O}_m, \quad \forall m \in \mathbb{M}. \tag{B.3}$$

onde $p_{Ji}$ é dado.

E no modelo ECJD as equações que correspondem as da seção 6.2 são:

$$T_{\sigma_{Ji}} - T_{Ji} \geq p_{Ji} - L_{Ji}z \quad \forall Ji \in \mathbb{O}_s, \quad \forall s \in \mathbb{M}, \tag{B.4}$$

$$\sum_{Ji \in \mathbb{O}_s} L_{Ji} = 1, \quad \forall s \in \mathbb{M}. \tag{B.5}$$

onde $p_{Ji}$ é dado e

$$L_{Ji} = \begin{cases} 1 & \text{se } Ji \text{ é a última operação do ciclo da máquina ;} \\ 0 & \text{caso contrário.} \end{cases}$$

Partindo do modelo JSRA, se chegará a restrição de não sobreposição do ECJD. Para isso a formulação do JSRA será relaxada, analisando apenas o par (Ji,Qj), onde $Qj$ é o sucessor imediato de $Ji$ na máquina, pressupondo um sequenciamento pré-estabelecido na máquina dado na resolução do modelo, que somente é possível graças à natureza combinatória de um problema de programação inteira

mista, e assim utilizando a transformação de $t$ em $T$ parte-se da equação (B.1), usando-se a equação (6.7) e encontra-se:

$$T_{Qj} + z \sum_{g=1,g\in\mathbb{O}_Q}^{j-1} O_{Qg} - T_{Ji} - z \sum_{g=1,g\in\mathbb{O}_J}^{i-1} O_{Jg} \geq p_{Ji} - k_{Ji,Qj} z$$

e manipulando esta equação tem-se:

$$\begin{aligned} T_{Qj} - T_{Ji} &\geq p_{Ji} - k_{Ji,Qj} z - z \sum_{g=1,g\in\mathbb{O}_Q}^{j-1} O_{Qg} + z \sum_{g=1,g\in\mathbb{O}_J}^{i-1} O_{Jg} \\ &= p_{Ji} - \left(k_{Ji,Qj} + \sum_{g=1,g\in\mathbb{O}_Q}^{j-1} O_{Qg} - \sum_{g=1,g\in\mathbb{O}_J}^{i-1} O_{Jg}\right) z \\ &= p_{Ji} - L_{Ji} z \end{aligned}$$

Pois sempre será possível encontrar $k_{Ji,Qj}$ tal que: $k_{Ji,Qj} + \sum_{g=1}^{j-1} O_{Qg} - \sum_{g=1}^{i-1} O_{Jg} = L_{Ji}$. Partindo do modelo ECJD, se chegará a restrição de não sobreposição do JSRA. Para isso será analisado o par (Ji,Qj), onde $Qj$ é o sucessor imediato de $Ji$ na máquina, pois, como já foi dito, o modelo ECJD pressupõe um sequenciamento pré-estabelecido na máquina dado na resolução do modelo, e assim partindo-se da equação (B.4), com $(\sigma_{Ji} = Qj)$, usando-se a equação (6.4) encontra-se:

$$t_{Qj} - z \sum_{g=1,g\in\mathbb{O}_Q}^{j-1} r_{Jg} - b - t_{Ji} + z \sum_{g=1,g\in\mathbb{O}_J}^{i-1} r_{Jg} + b \geq p_{Ji} - L_{Ji} z$$

e manipulando esta equação tem-se:

$$\begin{aligned} t_{Qj} - t_{Ji} &\geq p_{Ji} - L_{Ji} z + z \sum_{g=1,g\in\mathbb{O}_Q}^{j-1} r_{Jg} - z \sum_{g=1,g\in\mathbb{O}_J}^{i-1} r_{Jg} \\ &= p_{Ji} - \left(L_{Ji} - \sum_{g=1,g\in\mathbb{O}_Q}^{j-1} r_{Jg} + \sum_{g=1,g\in\mathbb{O}_J}^{i-1} r_{Jg}\right) z \\ &= p_{Ji} - k_{Ji,Qj}\, z \end{aligned}$$

Para se retirar a hipótese da existência de um sequenciamento pré-estabelecido na máquina, é necessário tomar um certo cuidado para que o problema não fique infactível, assim soma-se a equação B.1 da seguinte forma $(\forall Ji, Qj \in \mathbb{O}_s, \quad \forall s \in \mathbb{M})$:

$$
\begin{aligned}
t_{Qj} - t_{Ji} &\geq p_{Ji} - k(Ji, Qj)\, z \\
t_{Ji} - t_{Qj} &\geq p_{Qj} - k(Qj, Ji)\, z \\
\hline
0 &\geq p_{Ji} + p_{Qj} - (k(Ji, Qj) + k(Qj, Ji))\, z \\
\Rightarrow k(Ji, Qj) &+ k(Qj, Ji) \geq (p_{Ji} + p_{Qj}) \,/\, z
\end{aligned}
$$

Logo pode-se fazer $k(Ji, Qj) + k(Qj, Ji) = 1$ para $k(Ji, Qj) \in \mathbb{Z} \quad \forall Ji, Qj \in \mathbb{O}_m, \quad \forall m \in \mathbb{M}$.

Portanto uma solução factível $\gamma \in \Gamma_1$ também pertence ao conjunto $\Theta_1$, bem como uma solução factível $\theta \in \Theta_1$ também pertence ao conjunto $\Gamma_1$.

**Lema B.2** : *Os conjuntos $\Theta_2$ e $\Gamma_2$ são iguais.*

**Demonstração:** No modelo JSRA as equações que representam as restrições de altura de recorrência que correspondem as da seção 6.2 são:

$$t_{J1} - t_{Jn} \geq p_{Jn} - Hz \quad n = \text{card}(J) \quad \forall J \in \mathbb{O}_J, \tag{B.6}$$

$$t_{J1} - t_{Q1} \geq 1 - z \quad \forall J, Q \in \mathbb{J}. \tag{B.7}$$

E no modelo ECJD as restrições que correspondem as da seção 6.2 são:

$$T_{J1} - T_{Jn} \geq p_{Jn} - O_{Jn} z \quad n = card(\mathbb{O}_J) \quad \forall J \in \mathbb{J}, \tag{B.8}$$

$$\sum_{i=1; i \in \mathbb{O}_J}^{n} O_{Ji} \leq H \quad \forall J \in \mathbb{J}, \tag{B.9}$$

$$\sum_{i=1; i \in \mathbb{O}_J}^{n} O_{Ji} \geq 1 \quad \forall J \in \mathbb{J}, \tag{B.10}$$

$$T_{J1} - T_{Q1} \geq 1 - z \quad \forall J, Q \in \mathbb{J}. \tag{B.11}$$

onde

$$O_{Jn} = \begin{cases} 0 & \text{se } H - \sum_{i=1; i \in \mathbb{O}_J}^{n-1} O_{Ji} = 0; \\ 1 & \text{se } H - \sum_{i=1; i \in \mathbb{O}_J}^{n-1} O_{Ji} = 1; \\ 2 & \text{caso contrário;} \end{cases}$$

Partindo do modelo JSRA, se chegará a restrição (B.8) do ECJD. Assim partindo-se da equação (B.6), utilizando-se a transformação de $t$ em $T$ dada pela equação (6.7), encontra-se:

$$T_{J1} - T_{Jn} - z * \sum_{g=1, g \in \mathbb{O}_J}^{n-1} O_{Jg} \geq p_{Jn} - Hz$$

e manipulando esta equação , sendo $g \in \mathbb{O}_J$, tem-se:

$$\begin{aligned} T_{J1} - T_{Jn} &\geq p_{Jn} - H\,z + z \sum_{g=1}^{n-1} O_{Jg} \\ &= p_{Jn} - \left(H - \sum_{g=1}^{i-1} O_{Jg}\right) z \\ &= p_{Jn} - O_{Jn}\, z \end{aligned}$$

Da mesma forma partindo-se da equação (B.8), se chegará a restrição (B.6), assim usando-se a equação (6.4) encontra-se:

$$t_{J1} - b - t_{Jn} + z * \sum_{g=1, g \in \mathbb{O}_J}^{n-1} r_{Jg} + b \geq p_{Jn} - O_{Jn}\, z$$

e manipulando esta equação , sendo $g \in \mathbb{O}_J$, tem-se:

$$\begin{aligned} t_{J1} - t_{Jn} &\geq p_{Jn} - O_{Jn} z - z \sum_{g=1}^{n-1} r_{Jg} \\ &= p_{Jn} - \left(O_{Jn} + \sum_{g=1}^{n-1} r_{Jg}\right) z \\ &\geq p_{Jn} - Hz \end{aligned}$$

A restrição (B.7) garante que todos os *jobs* distintos sejam inicializados sempre dentro de um mesmo intervalo de tempo, equivalente ao ciclo.

Partindo da restrição (B.7) do modelo JSRA, se chegará a restrição (B.11) do ECJD. Assim utilizando a transformação de $t$ em $T$, isto é a equação (6.8) tem-se:

$$\begin{aligned} t_{J1} - b - t_{Q1} + b &\geq 1 - z \\ T_{J1} - T_{Q1} &\geq 1 - z \end{aligned}$$

Partindo da restrição (B.11) do modelo ECJD, se chegará a restrição (B.7) do JSRA. Assim utilizando mais uma vez a transformação de $T$ em $t$ tem-se:

$$\begin{aligned} t_{J1} - t_{Q1} &\geq 1 - z \\ t_{J1} - t_{Q1} &\geq 1 - z \end{aligned}$$

Precisa ser ressaltado ainda que neste trabalho definiu-se na seção 3.1 o valor de $L_{J1,Q1} = 1$, mas com uma simples verificação nota-se que para qualquer valor positivo $L_{J1,Q1}$ especificado a equivalência entre os modelos é satisfeita, exceto no caso de $L_{J1,Q1} = 0$, pois pela definição de $T$ tem-se $T < z$.

Portanto uma solução factível $\gamma \in \Gamma_2$ também pertence ao conjunto $\Theta_2$, bem como uma solução factível $\theta \in \Theta_2$ também pertence ao conjunto $\Gamma_2$.

**Lema B.3** : *Os conjuntos* $\Theta_1 \cap \Theta_3$ *e* $\Gamma_1 \cap \Gamma_3$ *são iguais.*

**Demonstração:** Pelo Lema B.1 tem-se que:

$$\Theta_1 = \Gamma_1.$$

Assim utilizando-se este resultado é necessário verificar a intersecção deste conjunto com as restrições de precedência do *job* e de limites de variáveis e a equivaência entre os modelos deste novo conjunto. Deve ser observado ainda que a relação de precedência é dada pelo roteiro de fabricação, portanto, a relação entre a primeira e a última operação do *job* não será estudada aqui.

No modelo JSRA as equações que representam as restrições de precedência do *job* e de limites de variáveis que correspondem as da seção 6.2 são:

$$t_{Jj} - t_{Ji} \geq p_{Ji} - h_{Ji}z \quad \forall i, j \in \mathbb{O}_J, \quad j \leq card(\mathbb{O}_J), \quad \forall J \in \mathbb{J}, \tag{B.12}$$

$$t_{Ji} \geq 0 \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}, \tag{B.13}$$

onde $p_{Ji}$ é dado.

O valor de $h_{Ji}$ é dado obedecendo as condições do problema definido no Capítulo 3, isto é, $h_{Ji} = 0$ para $i < card(J)$, $h_{Jn} = H$ para $n = card(J)$ $\forall J \in \mathbb{J}$.

E no modelo ECJD as equações que correspondem as da seção 6.2 são:

$$T_{s_{Ji}} - T_{Ji} \geq p_{Ji} - O_{Ji}z, \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}, \tag{B.14}$$

$$T_{Ji} \geq 0, \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}, \tag{B.15}$$

$$T_{Ji} < z, \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}, \tag{B.16}$$

$$O_{Ji} \in \{0,1,2\}. \tag{B.17}$$

onde $p_{Ji}$ é dado.

Utilizando-se as definições de variáveis dadas na seção 3.1.3 chega-se a:

$$O_{Ji} = r_{Ji} \quad \forall i \in \mathbb{O}_J, \quad \forall J \in \mathbb{J}.$$

Com relação às restrições de precedência, a equivalência entre as primeiras operações dos modelos é dada pelas equações 6.5 e 6.8.

Tendo-se como ponto de partida o modelo JSRA, se chegará à restrição (B.14) do ECJD. Assim partindo-se da equação (B.12), utilizando-se a transformação de $t$ em $T$ dada pela equação (6.7), encontra-se para :

$$T_{Jj} + z * \sum_{g=1, g\in\mathbb{O}_J}^{j-1} O_{Jg} - T_{Ji} - z * \sum_{g=1, g\in\mathbb{O}_J}^{i-1} O_{Jg} \geq p_{Ji} - h_{Ji}z$$

e manipulando esta equação , sendo $g \in \mathbb{O}_J$, tem-se:

$$\begin{aligned} T_{Jj} - T_{Ji} &\geq p_{Ji} - h_{Ji}\, z - z\sum_{g=1}^{j-1} O_{Jg} + z\sum_{g=1}^{i-1} O_{Jg} \\ &= p_{Ji} - \left(h_{Ji} + \sum_{g=1}^{i} O_{Jg} - \sum_{g=1}^{i-1} O_{Jg}\right) z \\ &= p_{Ji} - \left(h_{Ji} + O_{Ji}\right) z \\ &= p_{Ji} - O_{Ji}\, z \end{aligned}$$

Tendo-se como ponto de partida o modelo ECJD, se chegará a restrição (B.12) do JSRA. Assim partindo-se da equação (B.14), com ($s_{Ji} = Jj$), utilizando-se a transformação de $t$ em $T$ dada pela equação (6.4), encontra-se:

$$t_{Jj} - z * \sum_{g=1, g\in\mathbb{O}_J}^{j-1} r_{Jg} - b - t_{Ji} + z * \sum_{g=1, g\in\mathbb{O}_J}^{i-1} r_{Jg} + b \geq p_{Ji} - O_{Ji}z$$

- Um escalonamento $S$ atribui um tempo inicial $t(i,k) \in \mathbb{R}$ para cada ocorrência $< i,k >$.
- Um escalonamento $S$ é periódico com período $z$ se para cada operação genérica $i$, $t(i,k) = t_i + z*k$, onde $t_i = t(i,0)$. Note que o conjunto $S_g = \{t_i \in \mathbb{R}^+ \mid i \in T\}$, chamado de escalonamento genérico, e o período $z$ definem completamente um escalonamento periódico.

Pelas definições de Hanen (1994) e pelo Lema B.1 conclui-se que:

- Cada operação possui apenas um sucessor e um predecessor na máquina a que foi associada.

  É óbvio que a operação $i$ tem pelo menos um predecessor e um sucessor, pois no caso de $i$ ser a única operação atribuída à máquina, $i$ será o seu próprio sucessor e predecessor. Suponha-se por absurdo que uma operação $i$ tenha dois predecessores distintos $j$ e $h$ assim tem-se:

$$t(i,k_1) \geq t(j,k_2) \tag{B.22}$$

$$t(i,k_1) \geq t(h,k_3) \tag{B.23}$$

  Logo três alternativas são possíveis:

  1. $t(j,k_2) = t(h,k_3) \Rightarrow j = h$;
  2. $t(j,k_2) \geq t(h,k_3) \Rightarrow t(j,k_2) \geq t(h,k_3) + p_k$, porque senão haveria sobreposição de operações na máquina, o que não é possível como foi estabelecido no Lema B.1, portanto $j$ é o sucessor de $h$ e não de $i$;
  3. $t(h,k_3) \geq t(j,k_2) \Rightarrow t(h,k_3) \geq t(j,k_2) + p_j$, porque senão haveria sobreposição de operações na máquina, o que não é possível como foi estabelecido no Lema B.1, portanto $h$ é o sucessor de $j$ e não de $i$;

  Assim a existência de dois predecessores só será possível se $j = h$, o que contradiz a hipótese de $j \neq h$.

  O fato de não haver sobreposição de operações em uma máquina garante que qualquer operação $i$ não possui dois predecessores na máquina e com isso garante também que $i$ não possui mais que dois predecessores. Logo a operação $i$ tem exatamente um predecessor.

  Repetindo-se o procedimento acima usando-se a hipótese que a operação $i$ possui dois sucessores, conclui-se que a operação $i$ tem exatamente um sucessor.

  As restrições de unicidade de predecessor e de sucessor podem ser escritas da mesma forma que as equações B.19 e B.20.

- Não existem subseqüências cíclicas.

  Seja $S'$ uma subseqüência cíclica associada a máquina $m$ e $z$ o período associado ao escalonamento genérico

$S$. Cada operação $i$ associada a $S'$ satisfaz $t(i,k) = t_i + z * k$, $i \in \mathbb{O}_m$ e as restrições de unicidade de predecessor e sucessor. Temos duas possibilidades para $S'$:

- ou $S' = S$,
- ou $S' \subset S$.

Se $S'$ é o próprio $S$, então a subseqüência cíclica é a própria seqüência cíclica. Se $S'$ está propriamente contido em $S$, então existe pelo menos uma operação $j$ tal que , $j \notin S'$ e $j \in S$, pois a restrição de precedência garante a execução de todas as operações e pelo item anterior tem-se que as todas as operações dos *jobs* devem satisfazer as restrições de unicidade de predecessor e sucessor, assim ou a operação $j$ será inicializada após uma operação $i \in S'$ ou uma operação $i \in S'$ será inicializada após a operação $j$, mas isso significa que $j \in S'$ o que contradiz a hipótese de $j$, logo conclui-se que a única subseqüência cíclica $S'$ possível é a própria seqüência cíclica $S$, $S' = S$. Assim a não existência de subseqüências cíclicas também é uma característica do problema.

Esta restrição de não existência de subseqüências cíclicas poderia ser associada tanto a equação B.18 como a equação A.1 mostrada no Apêndice A.

- Cada máquina possui uma operação que pode ser designada como a última operação do ciclo.

  Como não existem subseqüências cíclicas e pela definição do escalonamento $S$ conclui-se que cada máquina possui uma operação que pode ser designada como última operação .

  Esta restrição pode ser associada a equação B.5.

Portanto uma solução factível $\gamma \in \Gamma_1 \cap \Gamma_3 \cap \Gamma_4$ também pertence ao conjunto $\Theta_1 \cap \Theta_3 \cap \Theta_4$, bem como uma solução factível $\theta \in \Theta_1 \cap \Theta_3 \cap \Theta_4$ também pertence ao conjunto $\Gamma_1 \cap \Gamma_3 \cap \Gamma_4$.

# C

# Restrições Adicionais

Para implementar as formulações discutidas neste trabalho, foi usado o *Software* **GAMS** (Sistema Genérico de Modelagem Algébrica), utilizando-se como *solver* o programa OSL. O GAMS/OSL é um pacote fechado com relação a forma de resolução das modelagens apresentadas, isto é, possui algoritmos próprios que não podem ser manipulados pelos usuários para resolver os problemas. Ele utiliza a estratégia de busca em árvores chamada *Branch-and-Bound*, **B & B**. A descrição desta metodologia é apresentada em detalhes na literatura pertinente à área de otimização e modelagem matemática.

Para inicializar a árvore, o **B & B** precisa de um limite inferior para a função objetivo, que é dado pela resolução do problema relaxado, onde todas as variáveis inteiras são consideradas contínuas.

No entanto, o limite inferior encontrado foi muito baixo para as formulações deste trabalho, criando, assim, uma necessidade de se 'fornecer' ao *solver* um limite inferior melhor.

Assim, duas inequações foram acrescentadas à formulação nos dois modelos:

- Como não existe sobreposição de operações nas máquinas, o valor do ciclo deverá ser maior que a soma dos tempos de processamento de todas as operações de uma máquina, para todas as máquinas, isto é:

$$\text{ciclo mínimo} = \text{máx}_{s \in \mathbb{M}} \{ \sum_{i \in \mathbb{O}_s} p_i \};$$

Assim será inserida nas duas formulações a seguinte restrição :

$$\sum_{Ji \in \mathbb{O}_s} p_{Ji} \leq Z \quad \forall s \in \mathbb{M}. \tag{C.1}$$

Para se mostrar que esta inequação é inerente ao problema, utilizando-se a restrição B.4, faz-se a seguinte soma de equações :

$$\begin{aligned}
T_{\sigma_{Ji}} - T_{Ji} &\geq p_{Ji} - L_{Ji}\, z \\
T_{\sigma_{\sigma_{Ji}}} - T_{\sigma_{Ji}} &\geq p_{\sigma_{Ji}} - L_{\sigma_{Ji}}\, z \\
&\vdots \\
T_{\sigma_{Qj}} - T_{Qj} &\geq p_{Qj} - L_{Qj}\, z \\
T_{Ji} - T_{\sigma_{Qj}} &\geq p_{\sigma_{Qj}} - L_{\sigma_{Qj}}\, z \\
\hline
\sum_{Ji \in \mathbb{O}_s} T_{Ji} - \sum_{Ji \in \mathbb{O}_s} T_{Ji} &\geq \sum_{Ji \in \mathbb{O}_s} p_{Ji} - z \sum_{Ji \in \mathbb{O}_s} L_{Ji}
\end{aligned}$$

e usando a equação B.5:

$$\begin{aligned}
0 &\geq \sum_{Ji \in \mathbb{O}_s} p_{Ji} - z \\
\Rightarrow z &\geq \sum_{Ji \in \mathbb{O}_s} p_{Ji}
\end{aligned}$$

Como as formulações são equivalentes pode-se acrescentar a equação C.1 aos modelos ECJD e JSRA sem restringir o espaço das soluções do problema.

- A altura de recorrência é usada para controlar o fluxo, e dessa forma tem que ser respeitada pelos dois modelos, assim será explicitada nas formulações a seguinte inequação :

$$\frac{H}{\sum_{i \in \mathbb{O}_J} t_i} \leq Z \quad \forall J \in \mathbb{J} \tag{C.2}$$