UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA QUÍMICA

ÁREA DE CONCENTRAÇÃO:
Sistemas de Processos Químicos e Informática

"Aplicação de Modelos MILP Baseados na Representação Contínua do Tempo em Problemas de Programação da Produção"

Autora: Kelly de Oliveira Cohen
Orientadora: Profª. Drª. Maria Teresa Moreira Rodrigues

Dissertação apresentada à Faculdade de Engenharia Química, como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Química.

Campinas, 02 de outubro de 1996

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

C66a

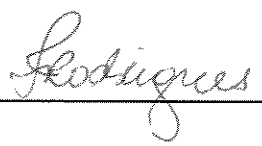
Cohen, Kelly de Oliveira
Aplicação de modelos MILP baseados na representação contínua do tempo em problemas de programação da produção / Kelly de Oliveira Cohen.--Campinas, SP: [s.n.], 1996.

Orientadora: Maria Teresa Moreira Rodrigues.
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Química.

1. Planejamento da produção. 2. Programação (Matemática). 3. Controle de produção. 4. Alocação de recursos. I. Rodrigues, Maria Teresa Moreira. II. Universidade Estadual de Campinas. Faculdade de Engenharia Química. III. Título.

Esta versão corresponde à redação final da tese de Mestrado defendida pela Engenheira Química Kelly de Oliveira Cohen e aprovada pela comissão julgadora em 02/10/96.

Orientadora



Profª. Drª. Maria Teresa Moreira Rodrigues

Tese defendida e aprovada em 02 de outubro de 1996, pela Banca Examinadora constituída pelos professores :

Profa. Dra. Maria Teresa Moreira Rodrigues

Prof. Dr. Roger Josef Zemp

Dr. Carlos Alberto dos Santos Passos

Agradecimentos

À Profª. Drª. *Maria Teresa Moreira Rodrigues*, pela orientação e dedicação durante este trabalho, e por tudo que me ensinou ;

Aos Prof. Dr. *Roger Josef Zemp* e *Carlos Alberto dos Santos Passos*, pela participação na Banca Examinadora ;

À CAPES, pelo apoio financeiro ;

À *Alcilene Monteiro*, *Cristina Drummont* e *Leila Zurba*, posso dizer que foram as primeiras amigas que encontrei em Campinas. Agradeço muito por tudo o que me ajudaram e pelo grande carinho que me deram ;

A *Edilson Santos*, *Luiz Rodrigues* e *Maria Marta Netto*, pelas valiosas sugestões neste trabalho, e principalmente por suas amizades e apoio. A eles tenho muito o que agradecer ;

A minha turma: *Cesar Carvalho*, *Daniele Casarin*, *Marco Fraga*, *Marta Reis*, *Paulo Reis*, *Reinaldo Silva*, *Ricardo Torres* e *Sergio Faria*, pelos momentos em que passamos juntos. Tenho certeza que jamais esquecerei desse tempo em que compartilhamos tanto os bons quanto os maus momentos. Tenho que enfatizar a sorte que tive de encontrar amigos tão companheiros ;

Agradeço também aos amigos *Frede Carvalho*, *Mila Torres*, *Silvia Casarin* e a todos os outros, perto ou longe, mas que sempre me ajudaram e a quem tenho muito carinho ;

Em especial ao meu namorado e amigo *Reinaldo*, que graças a realização de nossas teses nos conhecemos. E hoje posso dizer o quanto o amo ;

Aos meus pais *Nonato e Joana*, e minhas irmãs *Karla e Kayla*, pelo apoio e incentivo em todas as horas, e ao grande amor que nos une. A eles dedico este trabalho .

Quando sentimos confiança, fé e esperança que podemos concretizar nossos objetivos, isto constrói de nós um manancial de força, coragem e segurança.

Axline

Índice

Nomenclatura

- Índices utilizados para a formulação matemática do modelo :

u	Utilidades
s	Estado
i, i'	Tarefa
j, j'	Processador
k, k'	*Slot* de tempo
l, l'	Estágio
r	Recurso compartilhado
L_i	Estágios envolvidos na produção de **i**
L_j	Estágio correspondendo a unidade **j**
κ_j	Último *slot* de tempo definido para a unidade **j**
K_j	*Slots* de tempo postulados para a unidade **j**
l_j	Estágio correspondente a unidade **j**
$\overline{l_i}$	Estágio no qual a tarefa **i** é retirada
J_i	Grupo de unidades no qual podem processar a tarefa **i**
J_l	Grupo de unidades no qual pertencem ao estágio **l**

- Parâmetros utilizados :

EBT	Tempo de início mais cedo (*earliest beginning time*)
LFT	Tempo de fim mais tarde (*latest finish time*)
DD_i	Prazo de entrega das tarefas (*due date*)
RT	*Ready time*
HS_j	Instante de início do horizonte rolante
HF	Instante de fim do horizonte rolante
NU	Horizonte de controle
NY	Horizonte de predição
TP_{is}	Tempo de processamento para a saída de **i**
TP_{ij}	Tempo de processamento da tarefa **i** no processador **j**
U^{il}	Limitante superior (*Upper bound*)
SU_j	Tempo de transição dos processadores (*set-up*)
H_i	Pesos para as tarefas
CR_{ijr}	Quantidade de recurso **r** utilizada pela tarefa **i** na unidade **j**
OF	Oferta de recursos

- Variáveis :

Crt_{ij}	Criticalidade da operação
DW	Janela de demanda
W_{ijk}	Variável binária de alocação - aloca a tarefa **i** no processador **j** no *slot* **k**
B_{ijk}	*Batch sizes* (massa ou volume) produzidos no processador/equipamento **j** nos *slots* **k** e **k-TPij**
S_{sk}	Estoque no estado **s** no *slot* **k**

qe_{is}	Fração mássica ou volumétrica de entrada das operações **i** que produzem o estado **s**
qo_{is}	Fração mássica ou volumétrica de saída das operações **i** que produzem o estado **s**
D_{sk}	Quantidade de produto no estado **s** disponível para venda em **k**
R_{sk}	Quantidade de matéria prima no estado **s** disponível em **k**
C_s	Capacidade máxima de estocagem do produto **s**
Q_{uk}	Demanda da utilidade **u** no *slot* **k**
PP_{sk}	Preço da unidade de produto s no *slot* **k**
PU_{uk}	Preço da utilidade **u** no slot **k**
Vij^{min}	Capacidade mínima do processador
Vij^{max}	Capacidade Máxima do Processador
W_{ijkl}	Variável binária de alocação - aloca a tarefa **i** na unidade **j** pertencente ao *slot* de tempo **k** no estágio **l**
Y_{jk}	Variável de folga
TSS_{jk}	Tempo de início do *slot* **k** em **j**
TES_{jk}	Tempo de fim do *slot* **k** em **j**
TS_{il}	Tempo de início de **i** em **l**
TE_{il}	Tempo de fim de **i** em **l**
X_{ijl}	Variável binária de alocação - aloca a tarefa **i** no estágio **l** da unidade **j**
$ZA_{ili'l'}$	Variável lógica associada a $\delta S_{ili'l'}$ - $\delta S'_{ili'l'}$
$ZB_{ijli'l'}$	Variável lógica associada a $X_{ijl}.ZA_{ili'l'}$
$\delta S_{ili'l'}$	Variável binária associada a $TS_{i'l'}$ e TS_{il}
$\delta S'_{ili'l'}$	Variável binária associada a $TS_{i'l'}$ e TE_{il}
CON_{ilr}	Quantidade total do recurso **r** utilizado no início de produção da tarefa **i** no estágio **l**
Z_{il}	Variável positiva associada com adiantamento da tarefa
X_{il}	Variável positiva associada com o atraso da tarefa
S_{il}	Variável positiva e contínua, indica a existência ou não de estoque da tarefa/produto no estágio **l**
H_{il}	Variável positiva e contínua, indica o instante de tempo em que a operação anterior da rota esteja terminada
τ_{il}	Variável contínua associada com TS_{il}

Lista de Tabelas

Lista de Figuras

Resumo

Recentemente, os problemas de Programação da Produção em unidades químicas flexíveis tem recebido grande atenção na literatura, em função de sua importância econômica. Particularmente, a partir de 1993 tem surgido novas abordagens de modelamento do problema, com o objetivo de representar adequadamente as particularidades dos processos químicos, tais como reciclos .

No entanto, não existem ainda melhores abordagens capazes de resolver problemas de dimensão industrial. Neste sentido, foi proposto um modo de resolver tais problemas usando uma estratégia de horizonte rolante semelhante à empregada em problemas de controle preditivo. Nesta estratégia são criados subproblemas de menor dimensão do que o problema original, resolvidos seqüencialmente, existindo uma sobreposição parcial dos problemas sucessivos, de modo a garantir uma ligação mínima entre estes. Tais subproblemas podem ser resolvidos através de diferentes estratégias: Branch and Bound, heurísticas e programação matemática .

Neste trabalho foi estudada a possibilidade de usar uma abordagem de representação contínua do tempo, para modelar problemas de programação da produção dentro de uma estratégia de horizonte rolante. Foram propostas extensões da modelagem encontrada na literatura, de forma a adaptá-la aos problemas multipropósito normalmente existente na área de programação da produção .

Capítulo 1 - Introdução

Durante muitos anos, as operações contínuas ocuparam grande destaque na industria química. Porém, com o advento de novas especialidades químicas, os processos batelada passaram a despertar grande interesse econômico .

Uma das principais características dos processos batelada atuais é a flexibilidade, isto é, o mesmo conjunto de equipamentos pode ser empregado para produzir, em geral, baixos volumes de diferentes produtos, compartilhando não só os mesmos equipamentos, mas também outros recursos tais como utilidades, mão-de-obra, matérias-primas e intermediários. Como exemplos, pode-se citar as indústrias farmacêutica, de alimentos, corantes, aço, cosméticos, polímeros, bioquímica, etc .

Devido a flexibilidade de operação permitida por tais processos, tem-se uma nova categoria de problemas a ser resolvida, que podem ser estabelecidos como: Quais os instantes de tempo em que deve ser iniciada a produção de cada um dos produtos, de forma a:

- Atender a demanda;
- Satisfazer prazos de entrega;
- Satisfazer restrições de disponibilidade de matérias-primas;
- Satisfazer restrições sobre recursos compartilhados, incluindo equipamentos e demais recursos necessários à produção destes produtos;
- Satisfazer as restrições tecnológicas de produção (por exemplo, rotas de produção);

da maneira mais eficiente e/ou com um custo eficaz.

Em geral, a atividade de programação de curto prazo, também chamada *scheduling*, é precedida de uma atividade de Planejamento de Médio Prazo, que estabelece algumas metas a serem atingidas na programação de curto prazo, tais como prazos de entrega de produtos e quantidades a serem produzidas. No entanto, estas metas são em geral estabelecidas dentro de uma visão temporal agregada, que podem estabelecer metas difíceis de serem cumpridas. Além disso, atividades como manutenção, quebra de equipamentos, modificação da carteira de pedidos, podem exigir alteração do plano de produção. Assim, a programação de curto prazo deve ser flexível para suportar estas alterações .

A maior parte dos problemas de relevância industrial são de grandes dimensões, o que dificulta, ou mesmo impede, a obtenção da solução ótima dos problemas de *scheduling*, já que são classificados como NP-Completos (*Non-Polinomial Complet)*, isto é, o tempo de resolução destes problemas cresce exponencialmente com a dimensão do problema. Além disso são classificados como *hard problems*, o que os classifica como computacionalmente difíceis. Deste modo, a resolução de problemas de porte industrial somente é possível através da utilização de estratégias sub-ótimas de solução, principalmente estratégias heurísticas, as quais em geral não atendem critérios de otimização.

Para problemas de pequeno porte, existem várias estratégias de solução disponíveis na literatura, tais como busca em árvore e formulação matemática do problema. Nos últimos anos tem-se verificado grandes avanços na formulação matemática de problemas de *scheduling*, através de diferentes propostas de representação do tempo, tais como a discretização uniforme do tempo (Kondili et al. 1993) e a representação contínua do tempo (Pinto, 1995). Ambas as representações geram problemas do tipo MILP (*Mixed Integer Non-Linear Problem*), que podem ser resolvidos em geral apenas para pequenos problemas.

Recentemente, Latre et al. (1996) propuseram uma estratégia sub-ótima para resolver problemas de programação da produção de maiores dimensões. Foi proposto um procedimento iterativo, baseado em uma estratégia de horizonte rolante, no qual a cada passo é definido um subproblema do problema original. No procedimento proposto, o subproblema é formulado como um MILP baseado em uma representação discretizada do tempo. No entanto, a formulação utilizada, embora bastante geral, pode apresentar algumas limitações as quais, dependendo do caso, podem ser importantes.

Neste trabalho, o objetivo é explorar a formulação matemática baseada numa representação contínua do tempo, e propor alternativas na modelagem original de forma a adaptá-la à abordagem de horizonte rolante.

Capítulo 2 - Caracterização do Problema de Programação da Produção

2.1 - Introdução

No *scheduling* tem-se como objetivo determinar a carta temporal de eventos (Carta de *Gantt*) otimizando, se possível, um critério de desempenho da planta. Nesta carta é apresentado um programa detalhado que estabelece a ordem em que os produtos são produzidos e em que tempo os equipamentos devem ser alocados para a produção de cada batelada.

As principais características de um Problema de *Scheduling* e que afetam a escolha de uma estratégia de solução, são: estrutura de processamento, representação do tempo, critério de desempenho adotado, a presença ou ausência de armazenagem intermediária, tempos de transição (*set-up*) e prazos de entrega (*due dates*) .

Com o objetivo de resolver problemas de maior dimensão, Latre et al. (1996) propuseram uma estratégia sub-ótima de solução, na qual o problema original é decomposto em subproblemas menores, sendo estes formulados como problemas de programação mista (MILP) baseados numa representação discretizada do tempo. A decomposição do problema original é feita com base no estabelecimento de horizontes de tempo reduzidos, no qual o subproblema é resolvido. No entanto, os subproblemas guardam uma relação entre si na medida que, tal como na aplicação de estratégias de horizonte rolante em controle preditivo, há uma superposição no tempo dos subproblemas. Neste trabalho tem-se por objetivo explorar uma representação do tempo contínua, para futuramente, inseri-la em uma estratégia de horizonte rolante.

A idéia básica de se explorar uma forma alternativa de representação do tempo, diferente da utilizada até o momento, é estudar para quais tipos de problema a representação contínua do tempo possa ser vantajosa, já que cada uma delas apresenta diferentes limitações.

A escolha da representação do tempo tem um impacto muito grande sobre o modelamento do problema e a sua dificuldade de solução. Por isso, neste capítulo, serão apresentadas duas abordagens de representação do tempo: Representação Discretizada do Tempo (Kondili et al., 1993) e Representação Contínua do Tempo (Pinto, 1995), sendo apresentadas suas principais diferenças. No entanto, qualquer que seja a representação do tempo adotada, será, em geral, obtido um Problema de Programação Inteira Mista (MILP) de dimensões elevadas que podem impedir a obtenção de uma solução global em um intervalo de tempo razoável . Através da utilização de uma estratégia de horizonte rolante, o subproblema é reduzido e técnicas de programação matemática podem ser empregadas de forma eficiente.

Neste capítulo será utilizada a seguinte organização:

- Breve descrição geral dos problemas de programação da produção;
- Descrição sucinta das técnicas empregadas para a solução dos problemas de programação da produção utilizadas para a solução de problemas MILP;
- Apresentação da estratégia de horizonte rolante, principalmente com relação aos requisitos mínimos necessários que devem ser satisfeitos por uma abordagem de programação matemática;
- Apresentação das duas representações do tempo: representação discreta e contínua, suas principais características e limitações.

2.2 - Descrição Geral do Problema de *Scheduling*

Em Baker (1973), defini-se o problema de *scheduling* como a alocação temporal de recursos para realizar um conjunto de tarefas, ou seja, deseja-se determinar o instante em que cada tarefa deve ser realizada.

Os problemas de Planejamento e Programação da Produção são, em geral, abordados através da utilização de uma estrutura hierárquica, constituída de até três níveis hierárquicos (Rippin,1993):

- Planejamento de Longo Prazo, no qual normalmente são tomadas as decisões táticas; por exemplo expansão de linhas de produção, produção de novos produtos, etc;
- Planejamento de Médio Prazo, no qual são tomadas decisões de quantidades a serem produzidas, prazos de entrega, etc;
- Planejamento de Curto Prazo, onde é gerada a carta temporal de eventos (carta de *Gantt*) .

A fronteira entre estes níveis não é precisa, sendo que estes são em geral interdependentes.

Neste trabalho será abordado o Planejamento de Curto Prazo, supondo-se que num nível anterior foram definidos os produtos e as quantidades a serem produzidas, os prazos de entrega e os instantes a partir dos quais os produtos possam ter a produção iniciada em função, por exemplo, de disponibilidade de matérias-primas, paradas para manutenção etc.

O problema de programação de curto prazo (*scheduling*) que será abordado neste trabalho, tem definidos a *priori*:

- o número de produtos (tarefas) a serem produzidos ;

- receita de produção, onde são definidas as etapas de processamento (também chamadas operações) e o consumo de recursos ;
- os prazos de entrega dos produtos (*due dates*) ;
- a natureza da armazenagem intermediária e a política de processamento ;
- a oferta de recursos necessários à produção dos produtos (tarefas) ;
- tempo de processamento das tarefas (tempo em que as tarefas são executadas nos processadores) ;
- tempo de preparação dos equipamentos (*set-up*) ;
- tempo de transferência dos produtos entre os processadores (*changeover*);
- a natureza dos produtos intermediários ;
- critério de desempenho adotado para otimizar o sistema .

Os critérios de desempenho mais comumente adotados, são :

- *earliness*- minimização do adiantamento das tarefas ;
- *tardiness* - minimização do atraso das tarefas ;
- *makespan* - minimização do tempo total de conclusão das tarefas ;
- *flow time* - minimização do tempo de residência máximo das tarefas .

Estes critérios de desempenho, em geral, não atendem sozinhos a complexidade das soluções e decisões envolvidas na atividade de *scheduling*. No problema em questão, será utilizada uma função de custo que pondera o atraso na entrega do produto e a não-realização dos produtos especificados.

Algumas vezes, o problema de "*scheduling*" pode ser tratado como um problema de sequenciamento, isto é, basta determinar a seqüência com que cada processador será ocupado. Nos problemas de sequenciamento a definição do instante de início de execução de cada operação, e a sua alocação, é trivial, pois será o instante de tempo em que o processador for liberado pela operação que o ocupa, ou o instante em que a operação precedente for finalizada no processador

anterior. Este é o caso, por exemplo, da estrutura de processamento com fluxo unidirecional em que todos os produtos apresentam a mesma rota de produção e a ordem de execução dos produtos em cada equipamento é a mesma (*flow shop* com seqüências de permutação). Estruturas e rotas de processamento mais complexas, armazenagem intermediária limitada ou limitação na oferta de recursos compartilhados, tais como mão-de-obra, a decisão de alocação é muito mais complexa. Neste caso, não é possível a separação em problemas de sequenciamento e alocação, tratando-se de um problema de *scheduling* propriamente dito (Campos, 1993) . A seguir, serão apresentados alguns aspectos relevantes aos problemas de sequenciamento e *scheduling* .

2.3 - Aspectos Relevantes para o Problema de *Scheduling*

2.3.1 - Estrutura de Processamento

As estruturas de processamento são classificadas em duas categorias : único estágio e multiestágios .

2.3.1.1 - Único Estágio

Neste tipo de estrutura, todas as tarefas/produtos são executadas em apenas um único estágio, sendo subdivididas em :

- 1 estágio / 1 processador/equipamento

Cada tarefa é processada em um estágio, havendo somente um processador disponível .

Figura 2.1 - Estrutura de 1 estágio / 1 processador

- 1 estágio / M processadores

Neste caso, cada tarefa é processada em um único estágio, dispondo de **M** processadores. Por possuir somente um estágio e **M** processadores, estes encontram-se em paralelo. Tais processadores podem apresentar produtividades idênticas ou diferentes .

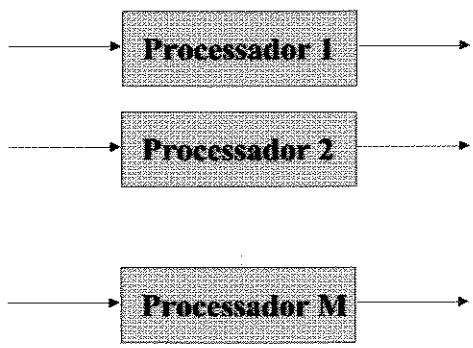


Figura 2.2 - Estrutura de 1 estágio / M processadores

2.3.1.2 - Multiestágios

Quando se deseja processar **N** tarefas em **L** estágios havendo **M** equipamentos disponíveis ($L \leq M$), tem-se as estruturas multiestágios. Este tipo de estrutura pode ser subdividida em:

- *Flow shop* (Multiestágios em Série)

Quando todas as tarefas seguem a mesma seqüência de processamento (fluxo unidirecional), denominada também de estrutura "multiproduto" .

Como visto anteriormente, existe um caso especial de *flow shop* conhecido como "*flow shop* com seqüências de permutação". Neste caso, a ordem de processamento das operações de cada tarefa é a mesma em todos os processadores.

Sendo assim, a solução do problema é dada em termos de seqüência de tarefas, pois não será necessário especificar a seqüência de operações para cada produto, já que é a mesma para todos eles .

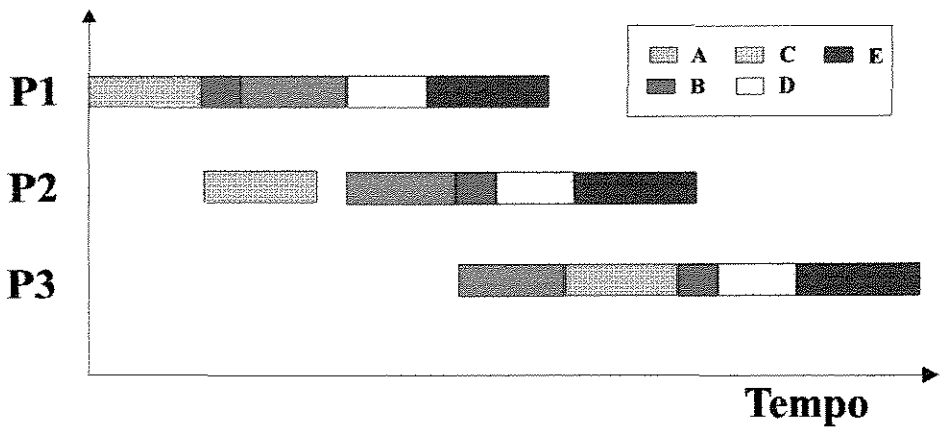


Figura 2.3 - Estrutura *Flow shop*

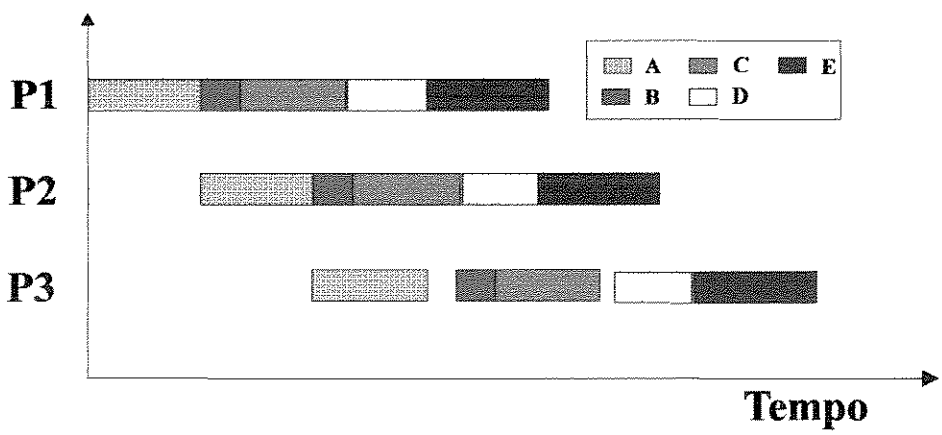


Figura 2.4 - *Flow shop* com seqüências de permutação

- *Job shop*

Quando os produtos possuem seqüências de processamento diferentes, podendo ser denominada de estrutura "multipropósito" (Figura 2.5) .

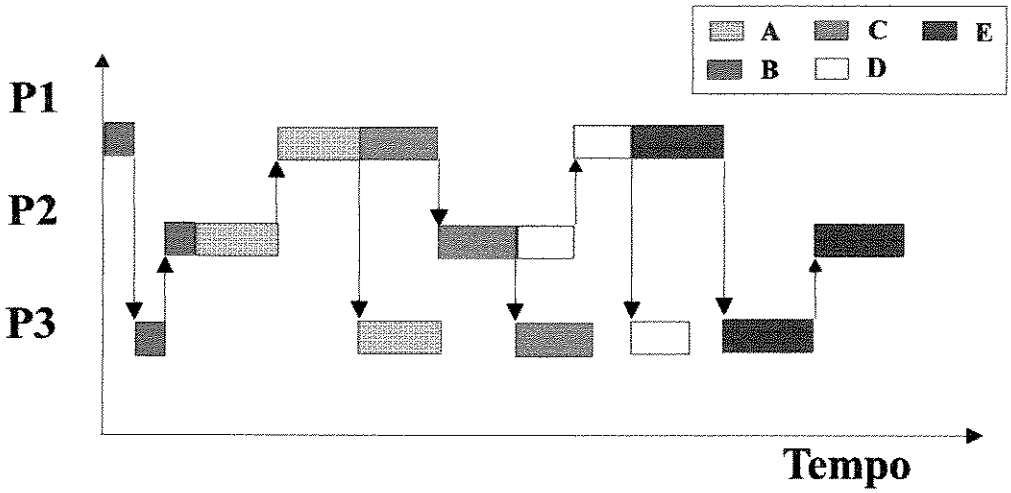


Figura 2.5 - Estrutura *Job shop*

2.3.2 - Armazenagem Intermediária

Armazenagem intermediária é definida como o estoque de produtos entre os estágios de processamento .

As principais razões para utilizá-la, são (Leinstein, 1990) :

- aumento da produtividade: a disponibilidade de tanques de armazenagem, permite que o processador seja liberado para processar outras tarefas quando o processador seguinte estiver ocupado ;
- facilita os *changeovers* ;
- nos processos semi-contínuos, permite que equipamentos adjacentes trabalhem a taxas de produção diferentes .

Dentre as políticas de armazenagem, tem-se (Baker, 1973; Karimi et al., 1988) :

- Armazenagem Dedicada: o tanque de armazenagem está associado a algum estágio de processamento, não sendo compartilhado entre os produtos intermediários ;

- Armazenagem Intermediária Infinita (*Unlimited Intermediate Storage* - UIS) : neste tipo de armazenagem, qualquer número de produtos em qualquer quantidade poderá ser atendido. É a política de armazenagem mais flexível ;
- Armazenagem Intermediária Finita (*Finite Intermediate Storage* - FIS) :os tanques de armazenagem estão disponíveis em quantidades limitada, ou seja, o material processado só poderá ser transferido se houver disponibilidade de tanque, caso contrário, permanecerá no mesmo até que possa ser transferido. Neste caso, a armazenagem poderá ser compartilhada entre estágios de processamento .
- Política de Processamento sem Armazenagem (*No Intermediate Storage* - NIS): não há armazenagem intermediária, sendo que o material processado poderá permanecer temporariamente no equipamento que o processou até que o próximo estágio seja liberado .
- Estrutura de Processamento sem Espera (*No Wait* - NW): não há armazenagem intermediária, sendo assim, o material que for processado deverá ser imediatamente transferido para o próximo estágio .

Pode ocorrer também a utilização de uma mistura destes tipos de política de armazenagem, dando origem a política *Mixed Intermediate Storage* (MIS) .

2.3.3 - Recursos Compartilhados

Quando as tarefas executadas simultaneamente requerem um determinado recurso, sendo este compartilhado entre as mesmas, introduz-se o conceito de recursos compartilhados. Dentre os recursos compartilhados, temos: equipamento, matérias primas, armazenagem, mão-de-obra e utilidades (vapor, água, eletricidade, etc.) .

Os recursos compartilhados apresentam características totalmente diferentes. Por exemplo, os equipamentos (ou processadores) são recursos do tipo 0/1, ou seja,

podem ser ocupados por apenas uma operação de um produto por vez. Comportamento semelhante tem a mão-de-obra. Já os recursos compartilhados do tipo energia elétrica, vapor e água, são recursos "capacitados", isto é, podem ser usados semelhantemente por várias operações, desde que o consumo total esteja abaixo do valor disponível .

2.4 - Tipos de Restrições

Em qualquer problema de *scheduling* estarão sempre presentes um conjunto de restrições que deverão ser obedecidas. Tais restrições atuam na dimensão do problema limitando o espaço de busca das soluções, mas, paradoxalmente, podem tornar a solução do problema mais difícil. Portanto, para uma solução eficiente do problema de *scheduling*, é necessário identificar e representar eficientemente as restrições do problema .

É importante notar que os problemas de *scheduling* tem, em geral, dois tipos de restrições: tecnológicas e de natureza econômica, como exemplo podemos citar :

- Restrições Tecnológicas: rotas de produção, interconectividade de equipamentos, dimensão de equipamentos e disponibilidade de matéria-prima;
- Restrições de Natureza Econômica: prazos de entrega e recursos compartilhados.

Estas restrições possuem normalmente graus de importância diferentes. Por isso, com respeito à importância relativa, as restrições podem ser classificadas em (Rodrigues, 1992):

- Restrições de infactibilidade ou restrições rígidas: são aquelas que infactibilizam um sequenciamento de tarefas e/ou alocação de recursos, e não aceitam relaxamento. Exemplo: restrições de precedência e restrições de armazenagem intermediária ;

- Restrições flexíveis: são aquelas que podem ser relaxadas, pois admitem certa folga e/ou a sua violação pode ser aceita, implicando em um aumento do custo total. Exemplo: prazo de entrega e recursos compartilhados .

Quanto a forma como as restrições interferem no processo de resolução, temos (Rodrigues, 1992) :

- Restrições estáticas: são restrições de infactibilidade que reduzem a *priori* o espaço de soluções factíveis. Exemplo: restrições de precedência ;
- Restrições dinâmicas: são aquelas que geram conflitos no instante em que deve ser tomada uma decisão de alocação. Exemplo: demanda simultânea de um recurso por uma ou mais tarefas .

2.5 - Metodologias Aplicadas para a Solução dos Problemas de *Scheduling*

Devido as dimensões e complexidade de grande parte dos problemas de *scheduling*, torna-se difícil a obtenção de soluções satisfatórias a qual possam obedecer todas as restrições presentes, por isso é necessário o uso de ferramentas adequadas que modelem o problema, diminuindo o espaço de soluções e viabilizando a sua otimização .

Existem basicamente três tipos de abordagens que são utilizadas para a solução dos diferentes problemas de Programação da Produção: Métodos de Busca, Programação Matemática, Regras e Algoritmos Heurísticos (Rodrigues, 1992) .

Para o objetivo deste trabalho, a ferramenta utilizada para a modelagem do problema de *scheduling* é a Programação Matemática .

2.5.1 - Técnicas de Programação Matemática

A utilização de técnicas de Programação Matemática depende da capacidade, e mesmo da possibilidade de representar matematicamente as restrições (de todos os tipos) e a função objetivo desejada .

Para a representação matemática adequada do modelo, é necessário frequentemente recorrer a variáveis "binárias" , isto é, que podem assumir exclusivamente valores zero ou um .

Esta característica leva a que os modelos de *scheduling* sejam de duas classes: modelos lineares mistos e modelos não lineares mistos, que conduzem a Problemas de Programação Linear Inteira Mista (MILP) e Programação não Linear Inteira Mista (MINLP) .

Existem alguns *softwares* para a solução de problemas do tipo MILP, tais como: OSL, LINDO e CPLEX; no entanto, para a solução de problemas do tipo MINLP, existe no GAMS o *software* DICOPT+ + , porém este não aceita funções não lineares em que figurem as variáveis binárias. Em função disto, resta praticamente como alternativa, linearizar as funções não lineares usando relações de dicotomia, e finalmente recorrer a um dos *softwares* para a solução de problemas do tipo MILP já citados .

Como já se disse, os problemas de *scheduling* são classificados , com respeito a sua natureza matemática, como: não-polinomiais (NP-completo) e "*hard*" (isto é, de difícil solução) . Estas características faziam com que o recurso à programação matemática fosse pouco usado. No entanto, com o desenvolvimento dos computadores, está havendo um esforço considerável no desenvolvimento de modelos matemáticos mais complexos para a representação do problema. Assim, antes de introduzir formulações mais recentes do problema de *scheduling* em plantas químicas, será apresentada a estratégia de solução típica de problemas de programação inteira mista .

2.5.1.1 - Programação Inteira Mista (Grossmann et al., 1992)

O planejamento e *scheduling* de processos bateladas são uma área muito fértil para a aplicação de técnicas de programação inteira mista. A razão para isto, é que muitos dos modelos de otimização matemática que surgem nestes problemas envolvem variáveis contínuas e discretas, que devem satisfazer a um conjunto de restrições de igualdade e desigualdade, e que deve ser escolhida uma função objetivo para otimizar (Grossmann et al., 1992) .

A forma mais geral de programação inteira mista corresponde ao problema de otimização :

Minimizar : $Z = f(x,y)$

Sujeito a :
$$\begin{cases} h(x,y) = 0 \\ g(x,y) \leq 0 \\ x \in R^n \ , \ y \in N_+^m \end{cases}$$

Onde **x** é um vetor de variáveis contínuas e **y** é um vetor de variáveis inteiras.

O problema acima, especializa-se em dois casos:

- Programação Linear Inteira Mista (MILP): Neste caso, a função objetivo **f** e as restrições **h** e **g** são lineares em **x** e **y**. Além disso, muitas das aplicações de interesse são restritas para o caso quando as variáveis inteiras **y** são binárias, ou seja, $y \in \{0,1\}^m$.
- Programação Não Linear Inteira Mista (MINLP): Neste caso, a função objetivo e/ou as restrições são não lineares. Aqui, a forma mais comum que tem sido assumida é linear nas variáveis inteiras e não linear nas variáveis contínuas .

A dificuldade que surge na solução dos problemas MILP e MINLP, é devido a natureza combinatorial desses problemas. Não há condições de otimalidade como nos casos contínuos, que possam ser diretamente explorados para o desenvolvimento eficiente de métodos de solução .

2.5.1.1.1 - Programação Linear Inteira Mista (MILP)

Na tentativa de desenvolver um método de resolução para o problema MILP, a primeira alternativa óbvia seria resolver cada combinação de variáveis binárias do problema PL correspondente em termos das variáveis contínuas, e

escolher como solução a combinação de variáveis binárias com menor função objetivo. O grande problema dessa abordagem, é que o número de combinações de variáveis binárias é exponencial, o que tornaria infactível computacionalmente problemas com grande número dessas variáveis .

Uma segunda alternativa é relaxar as variáveis binárias, e tratá-las como contínuas com limites $0 \leq y \leq 1$. Contudo, o problema desta abordagem, é que exceto para poucos casos especiais (exemplo: problema de designação de tarefas), não há garantia que as variáveis binárias terão valores inteiros na solução PL relaxada, não obtendo assim, a solução correta .

Desse modo, a solução de um problema MILP é constituída de duas etapas: (Baker, 1973)

- Resolver o problema relaxando as restrições sobre as variáveis binárias, utilizando a Programação Linear ;
- Proceder à busca da solução ótima satisfazendo as restrições de integralidade .

Para solucionar a segunda etapa do problema MILP, a estratégia normalmente empregada é a estratégia *Branch and Bound* (Ramificação e Sondagem).

O algoritmo *Branch and Bound*, consiste de dois procedimentos fundamentais (Baker, 1973) :

- *Branching* (Ramificação): é o processo, no qual um problema é dividido em dois ou mais subproblemas, ou seja, substitui-se o problema original por grupos de novos problemas. O resultado da ramificação é um conjunto de soluções parciais (chamadas nós) do problema original ;
- *Bounding* (Sondagem): é o processo em que é calculado o custo associado à solução parcial em questão, isto é, a cada nó.

O procedimento constitui uma enumeração parcial inteligente ou controlada dos grupos de soluções possíveis. A idéia básica é ramificar o problema em subproblemas, calculando-se a estimativa de custo associada a cada nó. Aqueles nós que apresentarem um custo superior a uma eventual solução conhecida ou obtida no processo de enumeração, pode ser eliminado, e este ramo não será mais explorado (Ku et al., 1987) .

- **Exemplo 2.1 :**

A seguir é apresentado um problema MILP envolvendo uma variável contínua e três variáveis binárias, o qual é resolvido utilizando-se a estratégia *branch and bound.*

Minimizar : $Z = x + y1 + 3y2 + 2y3$

Sujeito a : $-x + 3y1 + 2y2 + y3 \leq 0$

$-5y1 + 8y2 - 3y3 \leq -9$

$x \geq 0$, $y1,y2,y3 = 0,1$

A árvore *branch and bound* é mostrada na figura 2.6. O número nos círculos representam a ordem, no qual 9 dos 15 nós que representam as combinações possíveis da árvore são examinados para a achar a solução ótima. Note que a solução relaxada (nó 1), tem um *lower bound* de Z=5,8 e que a solução ótima encontrada no nó 9 tem Z=8, em y1=0, y2=y3=1 e x=3 .

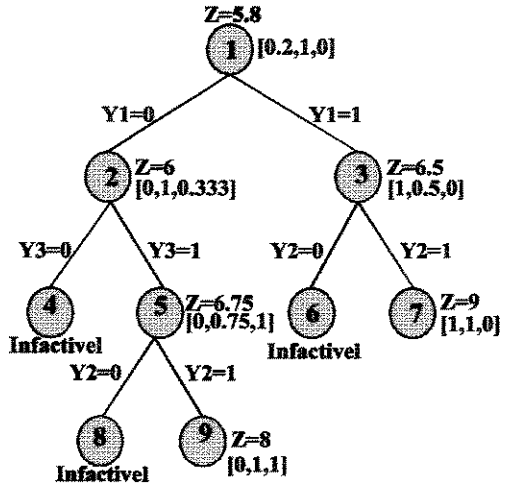


Figura 2.6: Árvore do *Branch and Bound*

2.6 - A Utilização do Horizonte Rolante como Estratégia Sub-ótima (Latre et al., 1996)

Freqüentemente, a dimensão e complexidade dos problemas impede a obtenção de uma solução ótima para o problema da programação de curto prazo. Como já foi dito, estes problemas são do tipo NP-completos, e portanto exigem que sejam utilizadas estratégias sub-ótimas para a obtenção de soluções. Além disso, quando o problema de *scheduling* considera os objetivos da produção e as restrições do processo ao mesmo tempo, torna-se difícil a sua solução. Consequentemente, são encontrados mais freqüentemente na literatura técnicas que utilizam somente ou os objetivos da produção ou as restrições do processo. Devido a esta dificuldade, foram propostos métodos de solução para reduzirem o espaço de soluções factíveis, facilitando a resolução do problema .

Uma das abordagens sub-ótimas possível de ser utilizada em problemas de Planejamento e Programação da Produção, é a estratégia de Horizonte Rolante (Latre et al., 1996) .

2.6.1 - Conceitos Básicos

O problema a ser tratado neste trabalho é :

Programar um conjunto de ordens com instantes de início (EBT_i) e prazos de entrega (DD_i) em uma planta multipropósito, sendo já definidas :

- Atribuição dos equipamentos a cada etapa de produção ;
- As quantidades a serem produzidas .

O problema de programação da produção estabelecido neste trabalho, tem como objetivo a obtenção de uma solução que represente um compromisso entre uma boa utilização da planta e o cumprimento dos prazos de entrega .

Como já foi dito, este problema é em geral impossível de ser resolvido de forma ótima, assim foi proposta uma estratégia para solucioná-lo de maneira sub-ótima. Nesta estratégia, o problema global é dividido em subproblemas resolvidos sucessivamente. Logo, é necessário estabelecer uma maneira de definir os subproblemas a serem resolvidos. A idéia básica proposta por Latre et al. (1996) é utilizar uma estratégia de horizonte rolante .

O horizonte rolante tem sido utilizado em técnicas de controle como uma estratégia sub-ótima para tratar o problema de controle, quando uma função de custo tem que ser minimizada em um horizonte de tempo, tal como no caso de controle preditivo. Na Figura 2.7 é apresentada, em linhas gerais, a utilização da estratégia de horizonte rolante no controle preditivo .

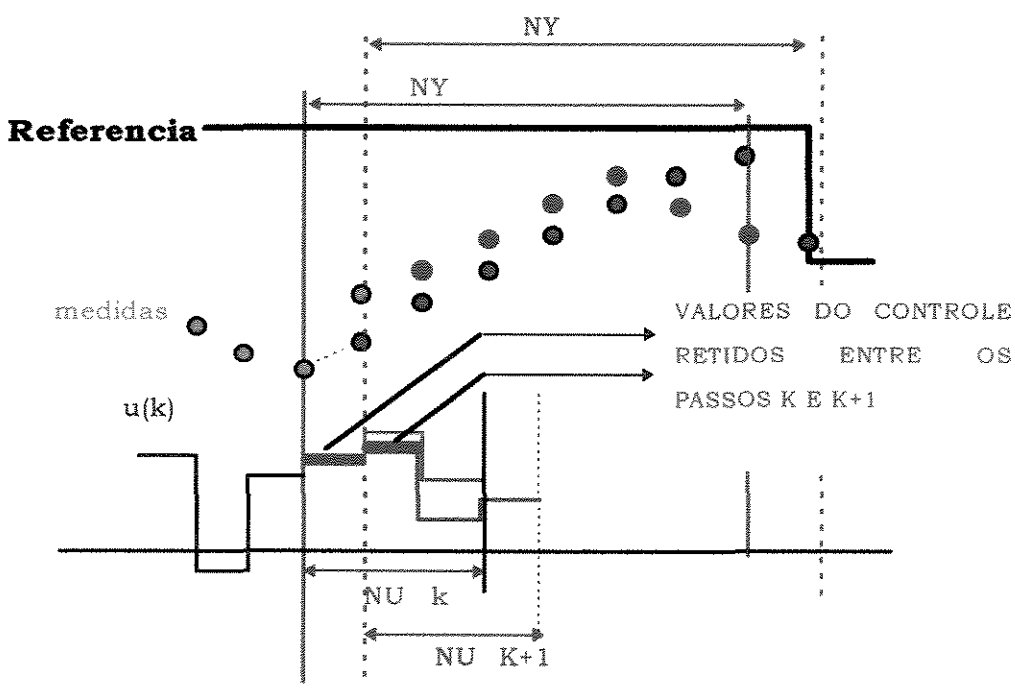


Figura 2.7 - Representação da utilização da estratégia de horizonte rolante no controle preditivo

São definidos horizontes de controle (NU), de predição (NY) e o modelo do processo. E a cada intervalo de tempo, tem-se :

- A saída é medida ;
- O modelo de predição é atualizado ;
- Os horizontes de controle e saída são avançados de um passo ;
- O valor do controle é calculado no intervalo NU, minimizando uma função de custo baseada nos erros entre o valor da referencia e a saída predita no horizonte NY ;
- Somente os valores de controle para o próximo intervalo são retidos .

A redução da complexidade do problema é obtida através de uma função de custo estendida apenas no horizonte de predição .

Esta estratégia foi adaptada para o problema de *scheduling* de curto prazo através da definição de um horizonte de alocação rolante, e um horizonte de predição (*lookahead*), baseado em estimativas de criticalidade das operações a serem realizadas (Keng et al., 1988) .

A utilização de uma estratégia de horizonte rolante tem dois objetivos principais :

1. Reduzir a dimensão do problema considerando um horizonte reduzido de alocação ;
2. Conferir um comportamento dinâmico ao problema de curto prazo, em função de permitir a aceitação contínua de ordens de produção, manutenção de equipamentos, etc .

Na Tabela 2.1 e Figura 2.8, faz-se um paralelo entre a estratégia de horizonte rolante no controle preditivo e a estratégia de horizonte rolante adaptada ao problema de *scheduling* .

Tabela 2.1: Adaptação do horizonte rolante ao problema de *scheduling*

Horizonte de controle	Intervalo de tempo para cada equipamento • iniciando-se quando o equipamento torna-se disponível ; • mesmo instante de fim para todos processadores
Saída do horizonte de predição	Conjunto de intervalos de tempo no futuro com alta contenção para os equipamentos
Variável de controle	Alocação de operações capazes de iniciarem no horizonte de controle
Predição da saída	Estimativa de conflitos (contenção)
Minimização de uma função quadrática do erro entre referencia e predição da saída	Minimização da soma dos pesos de : • contenção máxima ; • relaxamento dos prazos de entrega ; • tempos mortos no horizonte de alocação .
Somente o primeiro valor do controle é retido	Um subgrupo de decisões de alocação no passo é retido

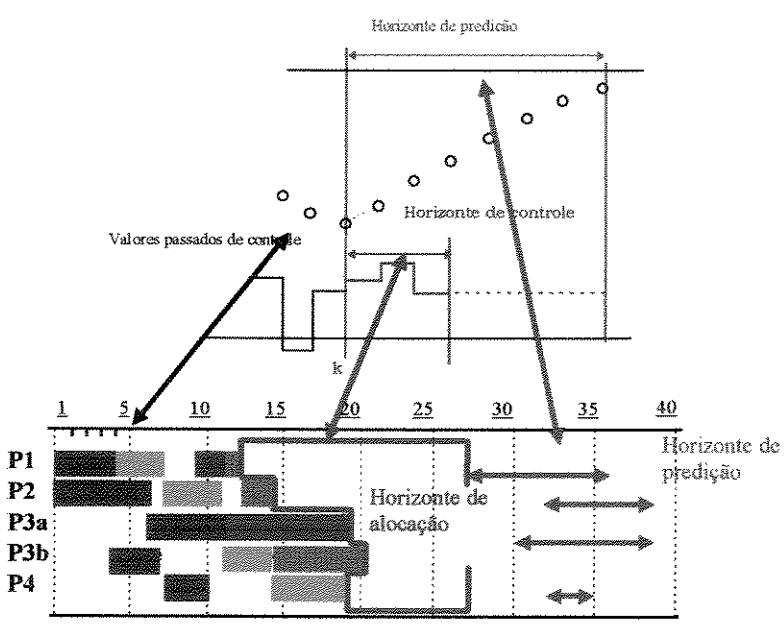


Figura 2.8 - Analogia entre os horizontes rolantes do controle preditivo e programação da produção

Entre dois passos sucessivos do algoritmo, os horizontes são atualizados havendo uma superposição parcial entre eles, já que nem todas as alocações decididas num determinado passo **k** são retidas para o passo **k+1.** Na Figura 2.9 é ilustrada a dinâmica da estratégia de horizonte rolante proposta por Latre et al. (1996) .

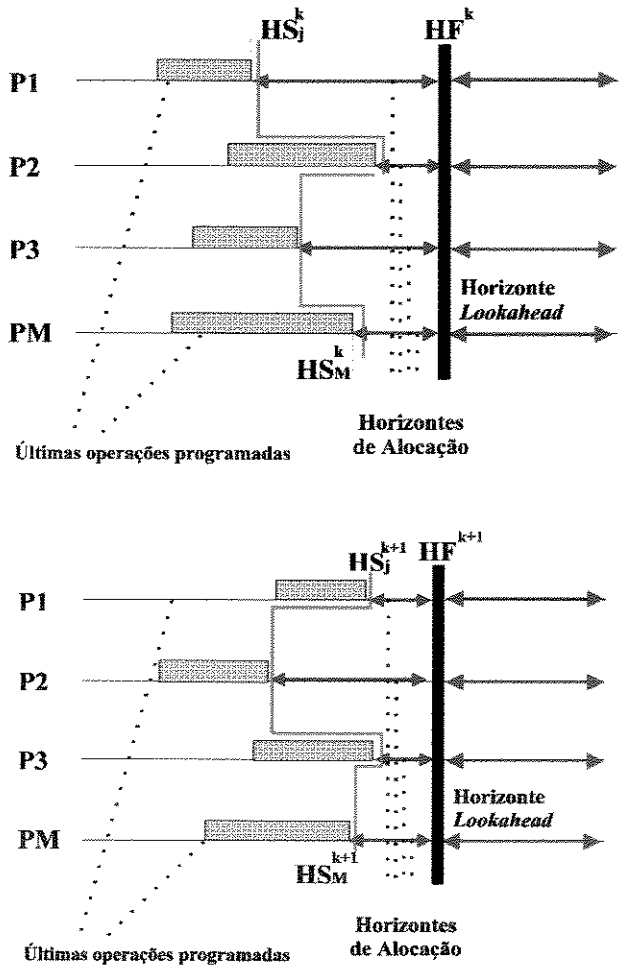


Figura 2.9: Unidades dos horizontes rolantes para alocações propostas e horizonte *lookahead*

Nesta figura, HS_j^k representa o início do horizonte rolante em cada processador **j** no passo **k**; HF^k representa o final do horizonte rolante neste mesmo passo. Uma vez definido os horizontes em cada processador (ou equipamento), segue-se uma estratégia de definição do subproblema a ser resolvido nestes

horizontes. Na abordagem proposta em Latre et al. (1996), a definição do subproblema é baseada nas janelas de demanda de cada produto e/ou operação. A seguir serão apresentados os conceitos de janela de demanda e a estratégia utilizada para definir os subproblemas .

2.6.2 - Janelas de Demanda

Em plantas reais, os produtos a serem produzidos tem definidos, em geral, os instantes de tempo mais cedo EBT_i a partir do qual seu processador pode ser iniciado, e o instante mais tarde (DD_i) em que estes produtos devem estar terminados (prazos de entrega) .

A definição externa de EBT_i e DD_i podem ser o resultado de: prazos de entrega (*due date*), disponibilidade de matérias-primas, decisões de gerenciamento da planta com respeito a preferencia de horários (com o tempo especificado para o processamento de tarefas), restrições de tempo na utilização de recursos, indisponibilidade de equipamentos devido a manutenção, etc (Latre et al., 1996) .

Estas duas quantidades definem então a janela de demanda dos produtos. Janela de demanda é um intervalo de tempo factível tecnologicamente para a execução das operações de cada tarefa. Os instantes de início ou abertura das janelas são limitados pelas restrições tecnológicas de precedência entre as operações de uma mesma tarefa. Os instantes de término ou fechamento das janelas são limitados pelo prazo de entrega ou um outro prazo definido pelo usuário, nos instantes mais tarde em que as operações devem estar terminadas, de modo a satisfazer o prazo estipulado e as restrições tecnológicas (Rodrigues, 1992) .

Cada janela de demanda (DW) é limitada pelo instante de início mais cedo (*earliest beginning time* - EBT) e pelo instante de fim mais tarde (*latest finish time* - DD_i) de uma operação .

$$EBT_{ij} = EBT_i + \sum_{l=p}^{j-1} TP_{il} \quad [2.1]$$

$$DD_{ij} = DD_i - \sum_{l=p}^{j+1} TP_{il} \quad [2.2]$$

A janela de tempo deverá ser sempre maior ou igual ao somatório dos tempos de processamento das operações da tarefa. Por isso, uma solução será infactível (inconsistência das restrições temporais), quando a sua janela de demanda for menor que o tempo necessário ao processamento de uma operação.

Na Figura 2.10 é mostrado um exemplo de janela de demanda de uma tarefa formada por três operações, cujos tempos de processamento estão apresentados na Tabela 2.2 , sendo seu prazo de entrega igual a 30 ($DD_i = 30$) .

Tabela 2.2: Tempos de processamento das operações

Operação	Tempo de processamento
1	5
2	3
3	7

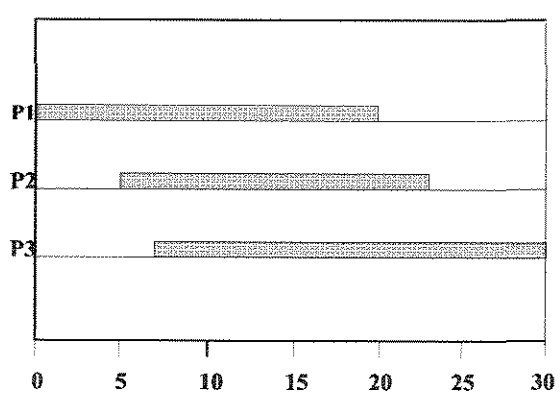


Figura 2.10: Exemplo de janela de demanda das operações de uma tarefa/produto para os dados da Tabela 2.2

É importante notar que cada produto tem uma janela de demanda inicial; no entanto, existe interdependência entre as janelas de demanda de operações de diferentes produtos, que requerem o mesmo equipamento para serem processados. Uma vez decidida a alocação de uma operação, haverá a propagação desta restrição, ocasionando o recorte, e consequentemente a diminuição das janelas de demanda de todas as operações que competem pelos mesmos equipamentos .

Com base nas janelas de demanda e os tempos de processamento de cada operação, Keng et al.(1988) definiram a criticalidade de uma operação :

$$Crit_{ij} = \frac{TP_{ij}}{DW_{ij}} \qquad [2.3]$$

Se a $Crit_{ij} > 1$, significa que será impossível a alocação de uma operação na sua janela de demanda .

A partir da criticalidade de cada operação, Keng et al. (1988) definiram a crucialidade de um intervalo de tempo como sendo :

$$(Crucialidade\ de\ um\ intervalo\ de\ tempo)_j = \sum_i Crit_{ij} \qquad [2.4]$$

A crucialidade do intervalo de tempo é então uma medida da contenção, isto é, da competição por um dado recurso e, consequentemente, é uma medida de gargalos no processamento. Quanto maior a crucialidade de um intervalo de tempo, maior é a competição por este intervalo, e maior é a contenção . A estratégia de *lookahead* tem então como objetivo fornecer subsídios para identificar gargalos no futuro, e assim identificar as operações que devem ser alocadas .

2.6.3 - Definição dos Subproblemas em Cada Passo

A cada passo do algoritmo são definidos :

1. O conjunto de operações que possuem $DD_i \leq HS^k$;
2. O conjunto de operações que, se não forem alocadas no passo **k**, terão uma janela de demanda inferior ao tempo de processamento ;
3. O conjunto de operações que tem $EBT \leq HS^k$ e $DD_i > HS^k$, e que contribuem para altas contenções no horizonte de predição .

A alocação destas operações é governada pela função de custo definida, em linhas gerais, como um compromisso entre :

- Alocação das operações especificadas em 1 e 2 , as quais, se não alocadas ocasionarão atraso na execução destas operações, e, consequentemente relaxamento dos prazos de entrega ;
- Minimização de gargalos futuros ;
- Boa utilização da planta .

$$\min f = \alpha \left\{ \begin{array}{l} \textit{Custo de atraso} \\ \textit{na execucao das} \\ \textit{operacoes do tipo} \\ \textit{1 e 2} \end{array} \right\} + \beta \left\{ \begin{array}{l} \textit{Custo de nao alocacao} \\ \textit{de operacoes que} \\ \textit{contribuem para} \\ \textit{gargalos no futuro} \end{array} \right\} + \left\{ \begin{array}{l} \textit{Custo da ociosidade} \\ \textit{dos equipamentos} \end{array} \right\}$$

O modelo do processo utilizado neste subproblema deve ter algumas características básicas :

- Permitir a alocação parcial de uma tarefa/produto, obedecidas as restrições de precedência. A alocação de operações é governada pala função de custo, portanto pode ocorrer situações em que as operações não sejam alocadas, o que significa que a variável binária de alocação seja zero ;
- Os prazos de entrega podem não serem satisfeitos, exigindo uma estratégia de relaxamento destes prazos .

2.7 - Formulação Matemática do Problema de *Scheduling*

As indústrias de processos químicos bateladas em unidades flexíveis, apresentam um conjunto de características que as diferenciam de outros tipos de processos flexíveis. Em geral, a estrutura de processamento é mais complexa, e geralmente são processados gases e líquidos, o que limita a interconectividade entre equipamentos. Porém, a maioria das estruturas de processamento apresentadas na literatura são bastante simplificadas, não levando em conta especificidades dos processos químicos. No entanto, é desejável que o modelo de produção seja capaz de atender a variações do fluxo de massa, instabilidade de intermediários, rotas alternativas (junção e separação de correntes), armazenagem intermediária, etc .

2.7.1 - Rede de Estados e Transições (*State Task-Network* - STN) (Kondili et al., 1993)

Os processos batelada envolvendo estruturas de produção complexas, podem ser representados usando uma rede de estados e transições, chamado de "*State Task-Network*" (STN). O novo aspecto dessa representação, é que tanto as operações de cada batelada individual, chamadas neste caso de tarefas, como as alimentações, produtos intermediários e finais, chamados de estados, são incluídos explicitamente como nós da rede (*network*). Processos envolvendo compartilhamento de matérias-primas e intermediários, junção e separação de fluxos e reciclos de material, podem ser representados, em quase todos os casos, sem ambigüidades.

2.7.1.1 - Representação do STN de Processos Químicos

O STN tem dois tipos de nós :

- Nós estados: representando as alimentações, intermediários e produtos finais ;

- Nós tarefas: representando as operações que devem ser realizadas para promover as mudanças de estado .

A representação STN é adequada para todos os tipos de estrutura de processamento (contínua, semicontínua ou batelada), podendo ser adaptada para qualquer tipo de formulação. As regras para sua construção são :

- Uma tarefa tem tantos estados de entrada (saída) quantos forem os tipos de material de entrada (saída);
- Duas ou mais correntes entrando no mesmo estado são necessariamente de mesma qualidade. Se a mistura de correntes diferentes está envolvida no processo, então esta operação deverá formar uma tarefa separada .

As receitas da rede são adequadas para estrutura de processamento serial. Porém, quando as estruturas tornam-se mais complexas, elas envolvem ambigüidade. Já o STN é livre de ambigüidade associado com a receita da rede (*recipe network*) .

Na representação STN a rota de produção é definida através da seqüência de estados estabelecidos na receita tecnológica, e dos equipamentos habilitados a promover a transição entre dois estados sucessivos, definidos na receita operacional.

Para ilustrar a diferença entre a representação de um processo através de um fluxograma tradicional e uma representação STN, apresenta-se na Figura 2.11 a representação de um processo através de um fluxograma tradicional, e nas Figuras 2.12 e 2.13 as representações STN que são possíveis de serem inferidas deste fluxograma.

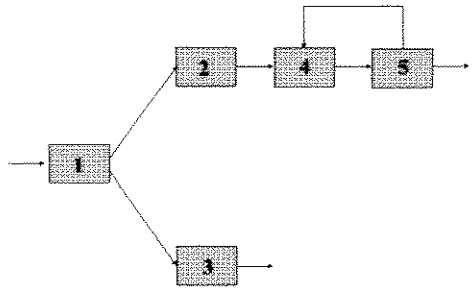
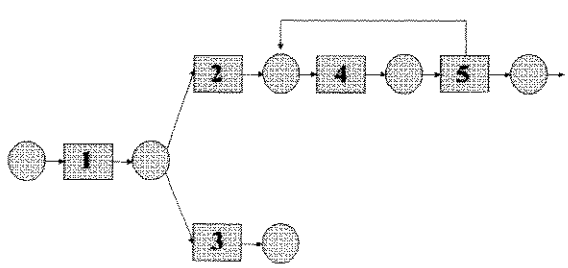


Figura 2.11: Fluxograma de um processo químico

Na Figura 2.11, não fica claro se a tarefa 1 produz dois diferentes produtos formando as entradas das tarefas 2 e 3, respectivamente, ou se ela tem somente um tipo de produto, no qual é compartilhado entre 2 e 3. Também, é impossível determinar se a tarefa 4 requer dois diferentes tipos de alimentações, produzidas pelas tarefas 2 e 5, respectivamente, ou se ela somente necessita de um tipo de alimentação, no qual pode ser produzida por 2 ou 5. Ambas as interpretações são possíveis .

Nas Figuras 2.12 e 2.13, são mostradas as duas representações STN, correspondentes ao fluxograma da figura 2.11 . As tarefas (transições entre estados), são representadas por □ e os estados (produtos intermediários, matérias-primas) por ○ .

A Figura 2.12 apresenta a tarefa 1 produzindo somente um intermediário, o qual é compartilhado com as tarefas 2 e 3. Também, a tarefa 4 apresenta somente uma alimentação, as quais são produzidas por 2 e 5. Já na Figura 2.13, mostra-se que a tarefa 1 produz dois intermediários diferentes formando as entradas da tarefa 2 e 3, respectivamente. A tarefa 4 tem duas alimentações diferentes, produzidas pelas tarefas 2 e 5, respectivamente .



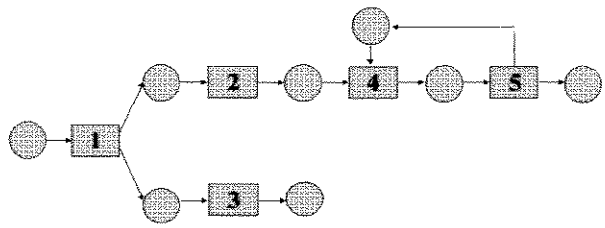


Figura 2.12 e 2.13: Representações do STN de processos químicos

Fazendo-se uma comparação entre a representação comum do fluxograma de um processo químico com a representação do STN, observa-se na segunda mais clareza e facilidade de entendimento do processo .

2.7.2 - Representação do Domínio do Tempo

Um aspecto fundamental para a resolução dos problemas de *scheduling* é quanto a representação do domínio do tempo. Há na literatura duas maneiras de representá-lo: Discretização Uniforme do Tempo (Kondili et al., 1993 ; Shah et al., 1993) e Domínio de Tempo Contínuo (Pinto, 1995) .

Os dois modelos que serão apresentados a seguir, utilizam como ferramenta de modelagem as técnicas da programação matemática, sendo formulados problemas do tipo MILP (Programação Linear Inteira Mista), apresentados na seção 2.5.1.1.1.

2.7.2.1 - Discretização Uniforme do Tempo (Kondili et al., 1993)

Na representação STN, uma operação pode ser realizada, se no instante de tempo **t** esteja disponível estoque dos estados necessários para a execução das tarefas. Além disso, em cada instante de tempo deve ser satisfeito o balanço material.

É claro que além da disponibilidade de material para a execução das tarefas, devem ser satisfeitas todas as demais restrições, como a ocupação dos processadores, etc . Por isso, a maneira mais simples de representar as condições citadas no domínio do tempo é através da discretização do tempo em intervalos regulares .

A formulação proposta por Kondili et al. (1993) baseia-se na discretização uniforme do tempo, sendo que o horizonte de tempo **H** é dividido em um número de intervalos de tempo (*slots*) de igual duração. O tamanho do *slot* de tempo é geralmente obtido com o máximo divisor comum (MDC) dos tempos de processamento envolvidos no problema. Eventos de qualquer espécie (início ou fim de processamento, variação na disponibilidade de equipamentos e outros recursos) devem ocorrer somente nos limites dos intervalos .

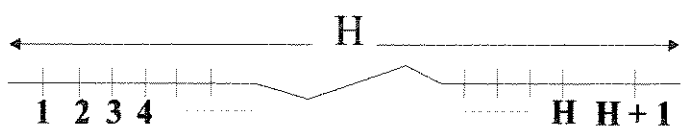


Figura 2.14: Discretização Uniforme do Horizonte de Tempo

As restrições fundamentais para serem satisfeitas são :

- Balanço material ;
- Resolução dos conflitos quando equipamentos são alocados para as tarefas;
- Limitações nas capacidades das unidades e armazenagens .

O modelo é composto pelas seguintes equações :

- **Balanço de massa**

O estoque de produto no estado s disponível no *slot* de tempo **k** será dado por:

$$S_{s,k} = S_{s,k-1} + \sum_{i \in To,s} qo_{i,s} \sum_{j \in Ki} B_{i,j,k-TPi,s} - \sum_{i \in Ti,s} qe_{i,s} \sum_{j \in Ki} B_{i,j,k,} - D_{s,k} + R_{s,k}$$

[2.5]

onde:

$S_{s,k}$ = estoque do estado **s** no *slot* **k**

$S_{s,k-1}$ = estoque do estado **s** no *slot* **k-1**

$B_{i,j,k-TPi,s}, B_{i,j,k}$ = *Batch sizes* (massa ou volume) produzidos no processador/equipamento **j** nos *slots* **k** e **k-TP** $_{i,s}$

$qo_{i,s}$ = fração mássica ou volumétrica de saída das operações **i** que produzem o estado **s**

$qe_{i,s}$ = fração mássica ou volumétrica de entrada das operações **i** que consomem o estado **s**

$D_{s,k}$ = Quantidade de estado **s** vendida no *slot* **k**

$R_{s,k}$ = Quantidade de estado s recebida no *slot* **k**

- **Equação de Não-Coexistência**

A variável binária Wijk é a variável de alocação, definida como :

$$W_{ijk} = \begin{cases} 1 \text{ se a tarefa } \mathbf{i} \text{ ocupa o processador } \mathbf{j} \text{ no } \textit{slot} \text{ de tempo } \mathbf{k} \\ 0 \text{ se caso contrario} \end{cases}$$

Assim, se terá:

$$\sum_{i} \sum_{k'=k}^{k-TPij+1} W_{ijk'} \leq 1 \qquad \forall j,k \qquad [2.6]$$

- **Restrições de capacidade**

A quantidade de material necessário para realizar a tarefa **i** no processador **j** no início do *slot* **k**, é limitada pelas capacidades mínima e máxima do processador:

$$W_{ijk} \cdot V_{ij}^{min} \leq B_{ijk} \leq W_{ijk} \cdot V_{ij}^{max} \qquad [2.7]$$

sendo: V_{ij}^{min} e V_{ij}^{max} as capacidades mínima e máxima, respectivamente, do processador **j** quando usado para realizar a tarefa **i** .

No caso em que as quantidades de material produzido ao longo de uma rota forem constantes (*batch sizes* constantes), a equação [2.6] pode ser simplificada, obtendo-se:

$$S_{s,k} = S_{s,k-1} + \sum_{i \in To,s} B_i \sum_{j \in Ki} W_{i,j,k-TPi,s} - \sum_{i \in Ti,s} B_i \sum_{j \in Ki} W_{i,j,k,} - D_{s,k} + R_{s,k} \qquad [2.8]$$

- **Restrição sobre a capacidade da armazenagem intermediária**

A quantidade de material armazenada no estado **s** não pode em nenhum instante de tempo **k** (*slot*) exceder a capacidade da armazenagem máxima .

$$0 \leq S_{s,k} \leq C_s \qquad [2.9]$$

onde: C_s = capacidade máxima de estocagem do produto **s**

- **Restrições sobre a oferta e consumo de recursos compartilhados**

As limitações sobre o consumo de recursos compartilhados, é dada por:

$$Q_{u,k} = \sum_{i} \sum_{j \in kj} \sum_{k'=k-TPi+1}^{k} C_{ij,u} W_{ijk'} \quad \forall k, u \qquad [2.10]$$

e a restrição sobre a limitação na oferta destes recursos é dada por:

$$Q_{u,k} \leq OF_{u,k} \qquad [2.11]$$

onde: $U_{u,k}$ = demanda da utilidade **u** no *slot* **k**

- **Função objetivo**

Neste caso os autores propuseram um critério de otimização em que o objetivo é a maximização do lucro, que é dado por:

$$\max \{ \sum_{s} PP_{s,H+1} S_{s,H+1} - \sum_{u} \sum_{k=1}^{H} PU_{u,k} Q_{u,k} + \sum_{s} \sum_{k=1}^{H} PP_{s,k} D_{s,k} - \sum_{s} \sum_{k=1}^{H} PP_{s,k} R_{s,k} \} \qquad [2.12]$$

onde:

$PP_{s,k}$ = preço da unidade **s** no *slot* **k** ;

$PU_{u,k}$ = preço da utilidade **u** no *slot* **k** .

2.7.2.2 - **Domínio de Tempo Contínuo** (Pinto, 1995)

Em Pinto (1995) é apresentada uma representação alternativa do tempo denominada "Domínio de Tempo Contínuo" (Representação não Uniforme do Tempo). Cada intervalo de tempo chamado de "*slot*" apresenta uma duração não definida a *priori*. Sua duração será definida pelo tempo de processamento da tarefa que ocupa o "*slot*". Na medida que a duração do "*slot*" é igual ao tempo de processamento da operação, então, no caso mais simples (sem processadores em paralelo), serão necessários tantos "*slots*" ou intervalos de tempo de duração variável quantas forem as operações a serem executadas no processador .

A modelagem apresentada por Kondili et al. (1993), tem como objetivo principal representar operações complexas presentes nos processos químicos, tais como junções, separações e reciclos de correntes, de modo que o ponto de partida é uma representação do tipo STN do processo. No caso da representação não uniforme do tempo, o objetivo foi inicialmente a representação de processos de produção mais simples, cujas principais hipóteses, são :

1. Demandas, prazos de entrega (*due date*), tempos de processamento, tempos de transição (*set-up*) e quantidades de recursos utilizados são fixos;
2. Cada tarefa deve ser processada por somente uma unidade em cada estágio, sendo as operações não preemptivas (não podem ser interrompidas durante o processamento) ;
3. Os tempos de transição são somente dependentes dos equipamentos ;
4. As quantidades produzidas em cada batelada de produção (*batch size*) são fixas, podendo haver, no entanto, várias bateladas de um mesmo produto, e a rota de produção é fixa .

As duas primeiras hipóteses são idênticas nas duas representações do tempo. A grande diferença está na hipótese de número 4, que fixa a rota e as quantidades produzidas. Na representação apresentada por Kondili et al. (1993) é equivalente

considerar B_{ijk} constante, obtendo-se imediatamente a equação de balanço de massa alternativa [2.8].

No caso da representação não uniforme, são utilizados dois domínios de tempo diferentes: o domínio do tempo das tarefas/produtos e o domínio dos processadores/equipamentos, que apresentam uma relação entre si. Estes domínios estão esquematicamente representados na Figura 2.15.

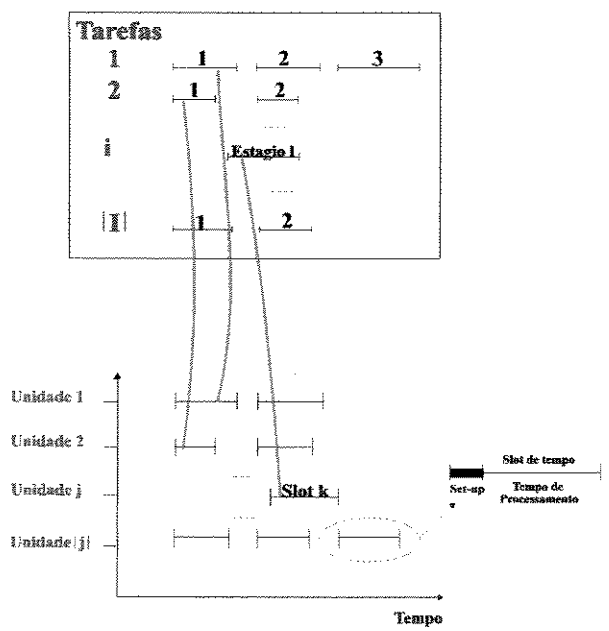


Figura 2.15: Coordenadas de tempo paralelas para a designação das tarefas aos equipamentos

Da Figura 2.15, tem-se que :

1. No domínio do tempo das tarefas são representadas as tarefas e suas respectivas relações de precedência tecnológica (seqüência de operações que compõe uma tarefa), de modo que os "*slots*" de tempo necessários à execução de uma tarefa não se sobreponham ;

2. No segundo domínio do tempo são representados os equipamentos e os "*slots*" disponíveis a atribuição de operações, devendo-se garantir que não haja sobreposição no tempo de "*slots*" consecutivos de um mesmo processador ;
3. Deve-se garantir que a cada instante de início de um "*slot*" no domínio das tarefas, corresponda a um instante de início do mesmo "*slot*" no domínio dos equipamentos ;
4. Consequentemente, cada "*slot*" pode ser ocupado, no máximo, por apenas uma operação em cada equipamento, definindo então a variável binária W_{ijkl}, onde :

$$W_{ijkl} = \begin{cases} 1 \text{ se a tarefa } \mathbf{i} \text{ ocupa o processador } \mathbf{j} \text{ no estagio } \mathbf{l} \text{ no } slot \text{ de tempo } \mathbf{k} \\ 0 \text{ se caso contrario} \end{cases}$$

Com base nas condições apresentadas acima, e tendo em conta que:

- são especificados os produtos **i** a serem produzidos ;
- a rota de produção de cada produto é representada pela seqüência de estágios de produção (o par produto/estágio representa uma operação) ;
- os equipamentos disponíveis à execução de cada tarefa em cada estágio ;
- os tempos de processamento .

foi proposto o seguinte modelo que segue :

- **Restrição de atribuição**

Cada operação de uma tarefa é executada apenas uma vez .

$$\sum_{j \in (ji \cap jl)} \sum_{k \in Kj} W_{ijkl} = 1 \qquad \forall i,l \in L_i \qquad [2.13]$$

- **Restrição de não-coexistência**

Cada *slot* de tempo em cada processador é ocupado por apenas uma operação.

$$\sum_{i \in Ij} \sum_{l \in (Li \cap Lj)} W_{ijkl} + Y_{jk} = 1 \qquad \forall j,k \in K_j \qquad [2.14]$$

onde: Y_{jk} é uma variável de folga, isto é, se não houver atribuição de uma operação a um *slot* .Em algumas situações , nenhuma tarefa é alocada para o *slot* de tempo (a variável de folga Y_{jk} torna-se igual a um); isto ocorre devido a uma superestimação do número de *slots* de tempo definidos para cada unidade .

- **Restrições de precedência tecnológica**

No domínio das tarefas, deve-se garantir a rota de produção, portanto :

$$TE_{il} \leq TS_{il+1} \qquad \forall i,l \in L_i - \{li\} \qquad [2.15]$$

e deve-se garantir que, uma vez iniciada uma operação ela terá reservado um tempo igual ao seu tempo de processamento:

$$TE_{il} = TS_{il} + \sum_{j \in (ji \cap jl)} \sum_{k \in Kj} W_{ijkl}(TP_{ij} + SU_j) \qquad \forall i,l \in L_i \qquad [2.16]$$

onde:

TS_{il} = tempo de início de **i** em **l**

TE_{il} = tempo de fim de **i** em **l**

TP_{ij} = tempo de processamento de **i** em **j**

SU_j = tempo de transições dos processadores

- **Restrições de alocação dos *slots***

No domínio dos equipamentos, deve-se garantir que não haja superposição de *slots* num mesmo equipamento, assim:

$$TES_{jk} \leq TSS_{jk+1} \qquad \forall j,k \in K_j - \{k_j\} \qquad [2.17]$$

Por outro lado, deve-se garantir que a duração de cada "*slot*" tenha a duração da operação a ele atribuída, então :

$$TES_{jk} = TSS_{jk} + \sum_{i \in Ij} \sum_{l \in Li} W_{ijkl}(TP_{ij} + SU_j) \qquad \forall j,k \in K_j \qquad [2.18]$$

onde :

TSS_{jk} = tempo de início do *slot* **k** em **j**

TES_{jk} = tempo de fim do *slot* **k** em **j**

A equação [2.19] garante *slot* de folga, ou seja, o *slot* de tempo **k+1** é usado somente no caso em que o *slot* **k** já tenha sido alocado . O principal objetivo desta restrição é reduzir o espaço de busca e a degeneração na otimização .

$$Y_{jk} \leq Y_{jk+1} \qquad \forall j,K \in k_j - \{k_j\} \qquad [2.19]$$

- **Equações de ligação entre os dois domínios (*clipping constraints*)**

A condição 3 estabelece que os instantes de início de cada *slot* em ambos os domínios deve ser idênticas (restrições de *clipping*), de modo que :

$$TS_{il} - TSS_{jk} \geq -U^{il}(1 - W_{ijkl}) \qquad \forall i,j \in (j_i \cap j_l),k \in Kj,l \in Li \qquad [2.20]$$

$$TS_{il} - TSS_{jk} \leq U^{il} (1 - W_{ijkl}) \qquad \forall i,j \in (j_i \cap j_l), k \in K_j, l \in L_i \qquad [2.21]$$

onde: U^{il} são constantes positivas grandes (*Upper Bound*). É desejável que se escolha um valor de U^{il} tão pequeno quanto possível para reduzir o espaço de busca, ou seja, eliminar soluções infactíveis, porém valores muito baixos de U^{il} podem eliminar soluções candidatas favoráveis .

- **Restrições de prazos de entrega**

$$TE_{il} \leq DD_i \qquad \forall i \qquad [2.22]$$

onde: DD_i = prazo de entrega da tarefa

- **Função objetivo**

O critério de desempenho adotado por Pinto (1995) é a minimização do *earliness*. O *Earliness* significa minimizar o intervalo de tempo entre o instante de tempo de fim do processamento das tarefas (TE_{il}) e seus prazos de entrega (*due dates* - DD_i), em outras palavras, ele visa reduzir indiretamente os custos de armazenagem dos produtos finais. Cada tarefa pode ser afetada de um coeficiente de ponderação H_i .

$$Min \sum_i H_i [DD_i - TE_{il}] \qquad [2.23]$$

2.7.2.2.1 - Modelagem com Restrições Sobre os Recursos Compartilhados

Neste trabalho, não se tem por objetivo tratar problemas com limitação sobre recursos compartilhados, no entanto a modelagem dos recursos compartilhados no domínio do tempo contínuo será apresentada apenas com o

objetivo de mostrar sua complexidade, em comparação com a modelagem dos recursos na abordagem de discretização uniforme do tempo.

O compartilhamento de recursos, tais como energia elétrica, vapor e mão-de-obra, podem ser de grande importância nos processos químicos. Se a oferta de tais recursos for limitada, isto é, se houver demanda simultânea e a possibilidade da demanda acumulada ultrapassar a oferta, então a alocação temporal das operações aos equipamentos deve ser feita tendo em conta tal limitação .

Como já se viu, no caso de se modelar o problema de *scheduling* através de uma abordagem de discretização uniforme do tempo, a limitação na oferta de recursos e o seu consumo é bastante simples, pois devido a definição de uma grade de tempo, não são acrescidas variáveis binárias ao modelo .

Na abordagem em questão, ou seja, na representação contínua do tempo, não existe uma grade temporal de referencia para modelar a simultaneidade da demanda. De fato, a restrição sobre recursos compartilhados impõe que a quantidade total de recurso **r** utilizado, deve ser menor ou igual a quantidade máxima de recurso ofertado .

$$CON_{ilr} \leq OF^{r} \qquad \forall i,l,r \qquad [2.24]$$

Pinto (1995) propõe a seguinte estratégia :

Considerando que o consumo de recursos é constante durante o tempo de processamento, e que a oferta de recursos é constante, é necessário apenas modelar os eventos notáveis, isto é, modelar na forma de degrau o instante de início da demanda e a sua duração, pois apenas nestes instantes ocorrem variações na demanda .

Na figura 2.16 é apresentada a idéia básica proposta pelo autor :

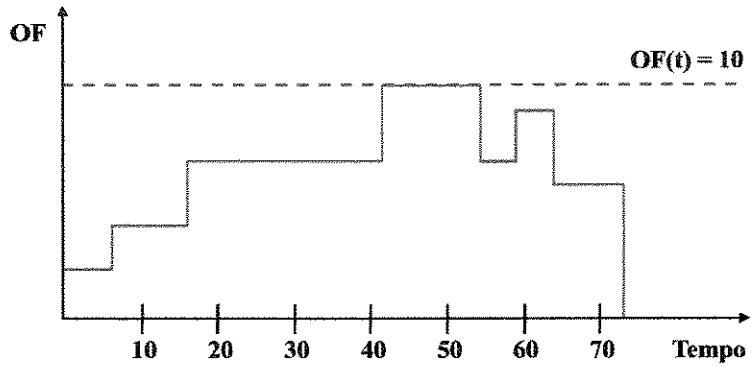


Figura 2.16 : Exemplo de perfil de consumo de recursos compartilhados

A demanda total de recurso em qualquer intervalo de tempo não pode exceder a disponibilidade máxima desse recurso. Dessa forma, é importante a sua variação sobre o horizonte de produção. O perfil de consumo dos recursos pode ser obtido reunindo a sua utilização sobre o tempo. Contudo, é somente necessário verificá-los nos instantes de tempo correspondentes ao início das operações, que são os instantes de tempo onde ocorrem mudanças nos consumos .

O perfil temporal de recursos pode então ser modelado como :

$$CONr(t) = \sum_{i} \sum_{j} \sum_{l} CR_{i,j,r} \; Xijl \; [\delta(t - TSil) - \delta(t - TEil)] \quad \forall r \qquad [2.25]$$

onde :

$\delta(t)$ é a função degrau dada por :

$$\delta(t) = \begin{cases} 1 \; se \; t \geq 0 \\ 0 \; se \; t < 0 \end{cases}$$

$CRijr$ = é a quantidade de recurso **r** usada pela tarefa **i** no processador **j**

$Xijl$ = variável binária de designação da tarefa **i** no estágio **l** na unidade **j**

Como já foi dito anteriormente, o cálculo do consumo de recurso se faz necessário somente nos instantes de tempo correspondentes ao tempo de início de execução das tarefas nos estágios. Desse modo, a equação [2.25] pode ser discretizada, como mostrado a seguir .

$$CON_{i'l'r} = \sum_{i} \sum_{j} \sum_{l} CR_{ijr} X_{ijl} [\delta (TS_{i'l'} - TS_{sil}) - \delta (TS_{i'l'} - TE_{il})] \quad \forall i', l \quad [2.26]$$

A equação acima possui termos não lineares, portanto uma linearização é realizada para substituir os termos não lineares da equação do consumo através de novas variáveis binárias. Primeiramente remove-se a função degrau δ(t) e são introduzidas as variáveis binárias que modelam os degraus $\delta S_{il'l'}$ e $\delta S'_{il'l'}$,através de relações de dicotomia. Este procedimento é detalhadamente apresentado em Pinto (1995), e conduz ao seguinte conjunto de inequações:

$$TS_{i'j'} - TS_{ij} \geq - U (1 - \delta S_{ijl'j'}) \quad [2.27]$$

$$TS_{i'j'} - TS_{ij} \leq U \, \delta S_{ijl'j'} - \varepsilon \quad [2.28]$$

$$TS_{i'j'} - TE_{ij} \geq - U (1 - \delta S'_{ijl'j'}) \quad [2.29]$$

$$TS_{i'j'} - TE_{ij} \leq U \, \delta S'_{ijl'j'} - \varepsilon \quad [2.30]$$

$$ZA_{il'l't} \geq \delta S_{il't} - \delta S'_{il't} \qquad \forall i,i',l,l' \quad [2.31]$$

$$ZA_{il'l't} \leq 1 - \delta S'_{il't} \qquad \forall i,i',l,l' \quad [2.32]$$

$$ZA_{il'l't} \leq \delta S_{il't} \qquad \forall i,i',l,l' \quad [2.33]$$

$$ZB_{ijli'l'} \geq X_{ijl} + ZA_{ili'l'} - 1 \qquad \forall\, i,j,i',l,l' \qquad [2.34]$$

$$ZB_{ijli'l'} \leq X_{ijl} \qquad \forall\, i,j,i',l,l' \qquad [2.35]$$

$$ZB_{ijli'l'} \leq ZA_{ili'l'} \qquad \forall\, i,j,i',l,l' \qquad [2.36]$$

onde:

ε = um parâmetro pequeno que pode ser fixado, por exemplo, em 0.01

$$\delta S_{iji'j'} = \begin{cases} 1 \text{ se } (i,j) \text{ precede } (i',j'), \text{ ou seja } TS_{i'j'} - TS_{ij} \geq 0 \\ 0 \text{ caso contrario} \end{cases}$$

$$\delta S'_{iji'j'} = \begin{cases} 1 \text{ se } TS_{i'j'} - TE_{ij} \geq 0 \\ 0 \text{ se caso contrario} \end{cases}$$

$ZA_{ili'l'}$ = variável lógica associada a $\delta S_{ili'l'} - \delta S'_{ili'l'}$

$ZB_{ijli'l'}$ = variável lógica associada a $X_{ijl}.ZA_{ili'l'}$

A quantidade total de recurso **r** utilizado no início de produção de uma tarefa **i** ' em um processador **j** ' é calculado por :

$$CON_{i'j'r} = \sum_i \sum_j CR_{ijr}\, ZB_{iji'j'} \qquad \forall\, i',l',r \qquad [2.37]$$

A introdução de restrições sobre os recursos compartilhados aumenta consideravelmente o tamanho do problema devido ao aumento de variáveis binárias, e consequentemente o tempo computacional requerido para sua solução .

2.7.3 - Diferença entre as Formulações

A principal diferença entre as formulações baseadas na representação uniforme e não uniforme do tempo, é com respeito à representação do domínio do tempo. Na primeira utiliza-se a discretização pré-definida do tempo. Neste caso, os *slots* possuem intervalos de tempo iguais. Já a segunda, trabalha com uma discretização implícita do tempo, ou seja, os *slots* não são pré-fixados .

Esta diferença básica tem como conseqüência um aumento bastante significativo com relação ao número de variáveis binárias necessárias à representação do problema.

Na formulação proposta por Kondili et al. (1993), o número de variáveis binárias é função da relação entre a dimensão do horizonte e a dimensão do "*slot*" de tempo, do número de tarefas/produtos e do número de equipamentos/processadores. A associação de uma grade de tempo simplifica a modelagem dos recursos compartilhados não acrescentando mais variáveis binárias ao problema.

Comparativamente, o número de variáveis binárias necessárias para a modelagem do problema no domínio do tempo contínuo também depende apenas do número de tarefas/produtos **i**, do número de processadores **j**, e do número de *slots* de tempo, **k**, no entanto, este último é função do número de tarefas/produtos que devem ser processadas em cada processador/equipamento. Em conseqüência, o número de variáveis binárias é bastante reduzido. Porém, o número de variáveis binárias aumenta quando há restrições sobre os recursos compartilhados, mas além disso, cresce computacionalmente a dificuldade de solução, pois originalmente o problema é não linear e a linearização é feita através da utilização de dicotomias, as quais, podem prejudicar o desempenho dos métodos de solução.

A seguir são apresentados alguns exemplos de aplicação das duas representações do tempo, com e sem recursos compartilhados.

- JM1 - Formulação com domínio de tempo contínuo, sem recursos;
- KOND1 - Formulação com discretização uniforme do tempo, sem recursos ;
- JM2 - Formulação com domínio de tempo contínuo, com recursos ;
- KOND2 - Formulação com discretização uniforme do tempo, com recursos .

Os testes foram realizadas em um PC/AT 486 DX4 100 MHz com 32Mb. Utilizou-se o *software* GAMS com o OSL como resolvedor. Usou-se como função objetivo a minimização do *Makespan*, que é uma medida do tempo total necessário para executar todas as operações.

Os dados do problema e os resultados computacionais estão ilustrados nas tabelas 2.3, 2.4 e 2.5 . A oferta de recursos é 14 (Latre et al., 1996) .

Tabela 2.3 - Processadores habilitados

Tarefas	Processadores
T_1	P_1,P_2,P_3
T_2	P_1,P_2,P_3
T_3	P_1,P_2,P_3
T_4	P_1,P_2,P_3

Tabela 2.4 - Tempo de processamento das tarefas e consumo de recursos

Tarefas	Tempo de processamento, TPij			Consumo de recursos		
	P_1	P_2	P_3	P_1	P_2	P_3
T_1	4	5	2	8	6	4
T_2	3	4	3	6	2	4
T_3	3	4	6	7	3	6
T_4	4	3	6	3	4	8

Tabela 2.5 - Resultados computacionais do problema :

	Sem Recursos		Com Recursos	
	JM1	KOND1	JM2	KOND2
Número de restrições	162	446	690	526
Nº total de variáveis contínuas	109	721	409	751
Nº de variáveis binárias	60	360	204	360
Solução relaxada	17.00	18.00	17.00	18.07
Solução inteira (*Makespan*)	24.00	24.00	30.00	25.00
OPTCR	0.01	0.01	0.01	0.01
Iterações	1196	12309	>100000	>100000
Nós	206	461		
Tempo (s. CPU)	15.21	168.18		

Pode-se observar na Tabela 2.5, que ambas as formulações com recursos não obtiveram solução satisfatória para os limites de tolerância dados. É certo que neste caso foram buscadas soluções com baixo *gap* de integralidade (OPTCR=0,01), muito abaixo do que é normalmente aceito (até cerca de 20%) .

A solução ótima encontrada é uma seqüência de permutação para JM1 e KOND1, ainda que o algoritmo não obrigue a isto. Suas soluções obtidas estão apresentadas nas Figuras 2.17 e 2.18.

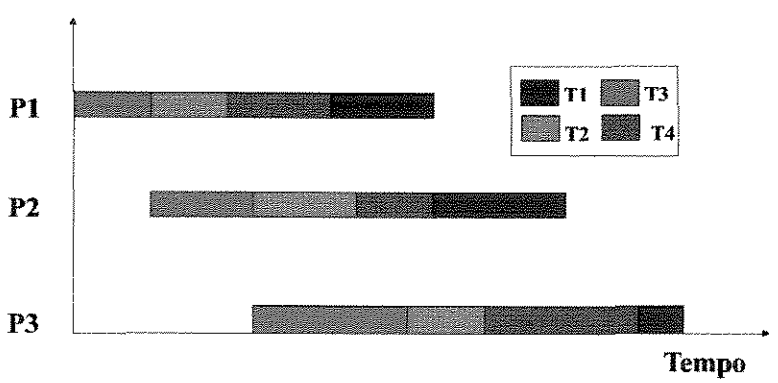


Figura 2.17 - Carta de *Gantt* de JM1

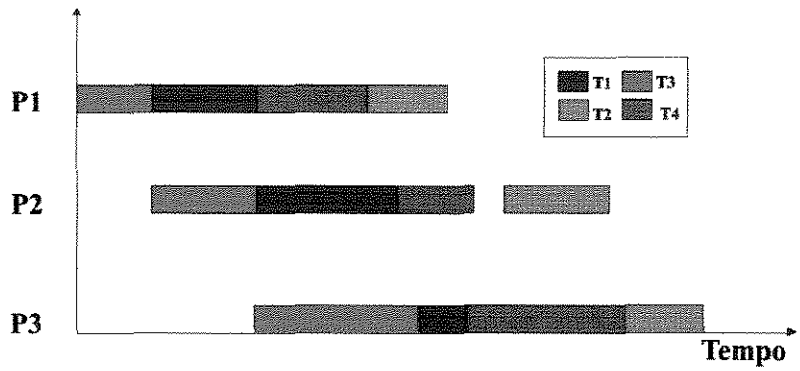


Figura 2.18 - Carta de *Gantt* de KOND1

Os resultados apresentados para este problema não são, é claro, conclusivos, pois as formulações foram aplicadas a apenas um exemplo. É certo que o desempenho de cada modelagem depende fortemente da função objetivo escolhida.

Cada uma das modelagens apresentada foi desenvolvida para atender uma classe diferente de problemas, tendo portanto características diferentes. A seguir são citados alguns aspectos relevantes de cada uma das formulações propostas, e algumas limitações identificadas.

- **Representação Discretizada do tempo**

Foi desenvolvida para atender problemas de programação de curto prazo em que não há operação cíclica da planta (Pantelides,1994). Na formulação original as quantidades a serem produzidas não são fixas e são determinadas em função de um critério de desempenho econômico da planta. Além disso, permite que o perfil de oferta de recursos seja variável no tempo. A associação de uma grade de tempo a representação STN permite a modelagem de sistemas bastante complexos, com reciclos, misturas de correntes, etc. A formulação apresentada permite também modelar processos com *set-up* dependente da seqüência, bem como modelar o consumo de recursos durante a operação de *set-up*. Uma das limitações é com respeito à modelagem da armazenagem intermediária. Embora possam ser modeladas todas as políticas de armazenagem (UIS, NIS, FIS, ZW), a

armazenagem, caso seja utilizada, ocupará no mínimo um *slot* de tempo. No caso do *slot* ser de grande duração, a armazenagem poderá ser utilizada por um período superior ao necessário.

- **Representação contínua do tempo**

Esta formulação, embora conceitualmente muito interessante, apresenta, em linhas gerais, maiores limitações do que a anterior. Como se viu, a dimensão das bateladas deve ser fixa, a modelagem de operações de *set-up* e preparação devem depender apenas do equipamento e não da seqüência, o consumo de recursos deve ser constante durante a realização das tarefas/produtos. No entanto, permite modelar facilmente vários tipos de armazenagem. A principal razão é que a utilização da armazenagem é um evento que depende das alocações, e não tem duração fixa, podendo eventualmente nem acontecer.

De fato as duas abordagens são bastante recentes, e ainda estão sendo exploradas para dimensionar seus potenciais. Em vista disto, este trabalho foi desenvolvido com o objetivo de estudar a adaptação da representação contínua do tempo a uma estratégia de horizonte rolante. Futuras alterações e adaptações podem ser exploradas mesmo porque não existem abordagens gerais para a solução de problemas de *scheduling*. É praticamente unânime na literatura especializada, que cada problema deve ser profundamente estudado de forma a se estabelecer a estratégia que melhor se adapte ao problema em estudo. Além destas restrições de caráter geral apresentadas neste capítulo, vão existir condições inerentes ao processo que devem sempre ser adicionadas dependendo do problema, que podem alterar significativamente o desempenho computacional na busca de uma solução.

Capítulo 3 - Proposta de Alteração da Formulação Baseada na Representação Contínua do Tempo

3.1 - Introdução

No Capítulo 2 foi apresentada a formulação proposta por Pinto (1995) (seção 2.7.2.2) e na seção 2.6.3 foram identificadas as características necessárias para que um modelo matemático possa ser utilizado para solucionar o subproblema de *scheduling* em uma planta química batelada, definido a cada iteração da estratégia de horizonte rolante.

As características necessárias são essencialmente duas:

a) Permitir o relaxamento de prazos de entrega, caso a solução ótima do subproblema assim o determine;
b) Permitir a alocação parcial de operações obedecidas as restrições tecnológicas.

Para atender às condições apresentadas acima, a formulação baseada em uma representação contínua do tempo proposta por Pinto (1995) precisa ser modificada.

Seja a formulação apresentada em 2.7.2.2 e reproduzida na Tabela 3.1, para o caso de não haver limitação em recursos compartilhados. A equação [2.22] estabelece que a solução obedeça os prazos de entrega; por outro lado a equação [2.13] estabelece que todas as operações de uma tarefa /produto sejam alocadas, obedecendo as restrições de rota tecnológica [2.15].

Para discutir e apresentar as limitações da formulação original, este capítulo será organizado da seguinte forma:

- Especificação de um exemplo que será utilizado para mostrar as limitações da formulação proposta ;
- Alteração da restrição [2.22] para permitir atrasos na entrega dos produtos finais;
- Alteração da restrição [2.13] para permitir a alocação parcial de tarefas garantindo a precedência tecnológica;

Tabela 3.1 - Formulação baseada na representação contínua do tempo

$$\sum_{j \in (ji \cap jl)} \sum_{k \in Kj} W_{ijkl} = 1 \qquad \forall i,l \in L_i \qquad [2.13]$$

$$\sum_{i \in Ij} \sum_{l \in (Li \cap Lj)} W_{ijkl} + Y_{jk} = 1 \qquad \forall j,k \in K_j \qquad [2.14]$$

$$TE_{il} \leq TS_{il+1} \qquad \forall i,l \in Li - \{li\} \qquad [2.15]$$

$$TE_{il} = TS_{il} + \sum_{j \in (ji \cap jl)} \sum_{k \in Kj} W_{ijkl}(TP_{ij} + SU_j) \qquad \forall i,l \in L_i \qquad [2.16]$$

$$TES_{jk} \leq TSS_{jk+1} \qquad \forall j,k \in Kj - \{kj \qquad [2.17]$$

$$TES_{jk} = TSS_{jk} + \sum_{i \in Ij} \sum_{l \in Li} W_{ijkl}(TP_{ij} + SU_j) \qquad \forall j,k \in K_j \qquad [2.18]$$

$$Y_{jk} \leq Y_{jk+1} \qquad \forall j,K \in kj - \{kj\} \qquad [2.19]$$

$$TSil - TSS_{jk} \geq -U^{il}(1 - W_{ijkl}) \qquad \forall i,j \in (ji \cap jl), k \in Kj, l \in Li \qquad [2.20]$$

$$TSil - TSS_{jk} \leq U^{il}(1 - W_{ijkl}) \qquad \forall i,j \in (ji \cap jl), k \in Kj, l \in Li \qquad [2.21]$$

$$TEil \leq DDi \qquad \forall i \qquad [2.22]$$

3.2 - Definição do Exemplo

O problema tratado neste trabalho é a obtenção de um *scheduling* de curto prazo para um grupo de tarefas. É suposto que no nível de planejamento já tenham

sido estabelecidos o número de bateladas a serem executadas e os prazos de entrega de cada produto, assim como as linhas de produção mais adequadas de acordo com as unidades disponíveis, contratos de entrega, inventário, etc .

A planta ilustrada na figura 3.1, possui uma estrutura de múltiplos estágios, totalizando quatro. Cada estágio possui um processador/equipamento, com exceção do terceiro estágio que contém dois processadores/equipamentos idênticos em paralelo .

Cada batelada de produto é uma tarefa, que por sua vez é decomposta em um grupo de operações ligadas através de restrições de precedência, que portanto definem as rotas de produção. Em geral, cada operação pode ser processada em mais de um equipamento, e cada tarefa possui um prazo de entrega. O fluxo é unidirecional e não há restrições sobre os recursos compartilhados .

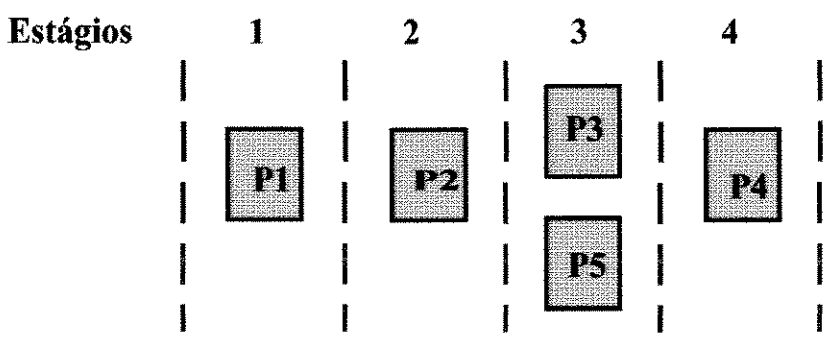


Figura 3.1: Descrição da planta

3.2.1 - Especificações da Planta e do Processo

- **Grupo de Tarefas/Produtos**

Há no problema um total de 5 tarefas, sendo estas compostas pelas seguintes operações (Tabela 3.2) :

Tabela 3.2 - Grupo de tarefas com suas respectivas operações

Tarefas	Operações
A	A_1,A_2,A_3,A_4
B	B_1,B_2,B_4
C	C_1,C_2,C_3
D	D_1,D_2,D_3
E	E_2,E_3,E_4

- **Equipamentos Disponíveis**

Estão disponíveis na unidade de produção 5 processadores, onde dois destes estão em paralelo (como visto na Figura 3.1) .

Como se trata de uma planta de múltiplos estágios, a tabela abaixo mostra os processadores disponíveis em cada estágio .

Tabela 3.3 - Processadores habilitados em cada estágios

Processadores	Estágios
P_1	L_1
P_2	L_2
P_3,P_5	L_3
P_4	L_4

O fluxo de produção é unidirecional, e as rotas de produção estão apresentadas na Tabela 3.4.

Tabela 3.4 - Rotas de Produção

Produto	**Operações**	**Processador**
A	A_1	P_1
	A_2	P_2
	A_3	P_3,P_5
	A_4	P_4
B	B_1	P_1
	B_2	P_2
	B_4	P_4
C	C_1	P_1
	C_2	P_2
	C_3	P_3,P_5
D	D_1	P_1
	D_2	P_2
	D_3	P_3,P_5
E	E_2	P_2
	E_3	P_3,P_5
	E_3	P_4

- **Determinação das Janelas de Tempo**

Cada produto/tarefa tem um instante de início mais cedo (*Earliest Beginning Time* - EBT) e um prazo de entrega (*Due Date*). Na Tabela 3.5 são apresentados estes dados, juntamente com os tempos de processamento de cada operação nos respectivos processadores/equipamentos .

Tabela 3.5 - Tempos de início, prazos de entrega e tempos de processamento das tarefas

Tarefas	EBT_i	Prazos de entrega	Tempo de processamento, TP_{ij}				
		Dd_i	P_1	P_2	P_3	P_4	P_5
A	0	25	3	4	3	5	3
B	0	30	5	2	0	2	0
C	0	20	1	2	6	0	6
D	0	20	3	2	5	0	5
E	0	15	0	4	3	3	3

3.3. - Restrições sobre os Prazos de Entrega

A formulação original de Pinto (1995) tem como função objetivo a minimização do *earliness*, isto é, a minimização entre o instante de término das tarefas/produtos e seus respectivos prazos de entrega, evitando-se o custo de armazenamento de produto final. Esta condição é mostrada na equação abaixo.

$$TEit \leq DDi \qquad \forall i \qquad [2.22]$$

A seguir é apresentada a aplicação da formulação proposta ao problema especificado em 3.2 .

Exemplo 3.1 :

Considere a planta da Figura 3.1 e os dados das Tabelas 3.2 a 3.5. O problema de otimização, é a minimização do "*earliness*", dado pela função de custo [2.23] e a formulação apresentada na Tabela 3.1.

Na Tabela 3.6 e na Figura 3.2 são mostrados os resultados computacionais e a carta de *Gantt* representando a solução ótima, respectivamente .

Tabela 3.6 - Resultados computacionais do Exemplo 3.1 :

V. Binárias Nº Total de Variáveis Nº de Restrições	PL Relaxado	Solução Inteira (Ealiness)	OPTCR	Nós	Iterações	Tempo (s. CPU)
125 208 339	0.00	0.00	0.01	7249.00	65983.00	859.53

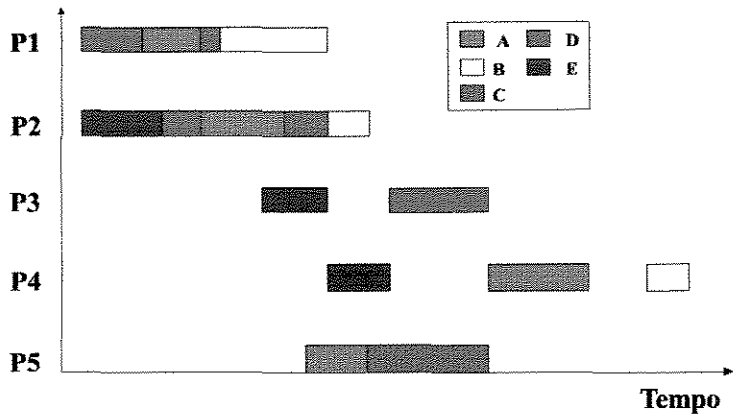


Figura 3.2 - Carta de *Gantt* do Exemplo 3.1

Analisando o problema do ponto de vista da função de custo proposta, o problema apresentará solução factível se todos os prazos de entrega forem factíveis. Em problemas reais, os prazos de entrega são definidos em um nível de planejamento onde se tem uma visão agregada do problema, podendo definir prazos de entrega impossíveis de serem cumpridos. Além disso, podem acontecer imprevistos, tais como quebras de equipamentos, indisponibilidade de matérias-primas, que inviabilizem o cumprimento dos prazos acordados. Assim, é interessante dispor de um modelamento que resolva o problema usando uma estratégia diferente, já que a apresentada por Pinto (1995) somente apontará o problema como *infactível*.

A alteração do prazo de entrega do produto A de 25 para 10, por exemplo, já torna o problema infactível.

A retirada da restrição 2.22 do modelo não é recomendável, na medida que altera significativamente o espaço de soluções do problema; por outro lado, é necessário dispor de uma estratégia que penalize apenas o atraso e não o eventual adiantamento na execução das tarefas.

Na equação 3.1 é apresentada a reformulação desta restrição de forma a calcular o atraso, no caso deste acontecer.

$$TE_{il} - DD_i + Z_{il} - X_{il} = 0 \qquad \forall i, l \in L_i \qquad [3.1]$$

onde :

Z_{il} = variável positiva associada com o adiantamento da tarefa

X_{il} = variável positiva associada com o atraso da tarefa

A equação [3.1] não atua como uma restrição formal do modelamento como as demais, apenas fornece subsídios para o cálculo do atraso X_{il}, se evidentemente houver atraso. É importante notar que X_{il} é uma variável positiva, de tal modo que o atraso poderá ser minimizado através de uma função de custo linear. Na Figura 3.3 são ilustradas as opções que podem acontecer na solução do problema.

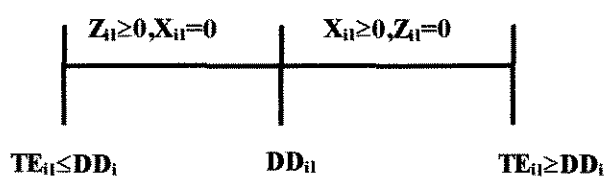


Figura 3.3 - Ilustração da restrição 3.1

1. Caso haja adiantamento das tarefas ($TE_{il} \leq DD_i$), Z_{il} será maior que zero e X_{il} será igual a zero ;
2. Se os tempos de fim de processamento das tarefas forem iguais aos prazos de entrega ($TE_{il} = DD_i$) , Z_{il} e X_{il} serão iguais a zero ;
3. Se houver atraso das tarefas ($TE_{il} \geq DD_i$), X_{il} será maior que zero e Z_{il} será igual a zero.

A introdução de uma condição tal como a [3.1], ainda permite otimizar o problema minimizando o *earliness*, bastando para tanto rescrever a função objetivo [2.21], tal como segue:

$$\min Earl = \sum_i \sum_l X_{il} + \sum_i \sum_l Z_{il} \qquad [3.2]$$

A seguir é apresentada a aplicação desta nova formulação ao exemplo apresentado na seção 3.1, alterando-se apenas o prazo de entrega do produto A de 25 para 10.

3.4 - Restrições de Alocação

A alocação parcial de uma tarefa/produto exige que:

- A restrição [2.13] seja relaxada, permitindo, se necessário, a não alocação da tarefa/produto no estágio, isto é:

$$\sum_{j \in (Ji \cap Jl)} \sum_{k \in Kj} W_{ijkl} \leq 1 \qquad \forall i,l \in L_i \qquad [3.3]$$

- As operações que forem alocadas devem obedecer a rota de produção.

No caso da formulação proposta, a garantia de obediência à rota tecnológica é obtida através da restrição:

$$TE_{il} \leq TS_{il+1} \qquad \forall i,l \in L_i - \{l_i\} \qquad [2.15]$$

No entanto, usando-se a restrição de alocação como em [3.3] e a restrição de rota como em [2.15], não é garantida uma solução tecnologicamente factível. Garante-se apenas que as operações alocadas de um produto/tarefa serão realizadas num instante de tempo posterior ao instante de término da operação da rota, não

importando se ela está ou não alocada. Isto é, se uma operação de uma tarefa/produto não for alocada, mesmo assim, por força das restrições [2.15] e [2.16] reproduzida abaixo:

$$TE_{il} = TS_{il} + \sum_{j \in (ji \cap jl)} \sum_{k \in Kj} W_{ijkl}(TP_{ij} + SU_j) \qquad \forall i,l \in L_i \qquad [2.16]$$

a ela terá associado um valor de $TE_{il} = TS_{il}$ apenas dependente do instante de término da operação anterior, que pode eventualmente ser zero, mas não ***se*** a operação anterior foi alocada .

Suponha por exemplo, para um produto A qualquer cuja rota de produção seja formada por 2 operações. Se a primeira operação não for alocada se terá:

$$W_{Ajk1} = 0 \ , \ TE_{A1} = TS_{A1} = 0$$

e pode ocorrer que:

$W_{Ajk2} = 1$ em qualquer instante de tempo, já que qualquer instante de tempo obedece a restrição $TE_{A1} \leq TS_{A2}$

No caso da abordagem via discretização uniforme do tempo, estas condições são garantidas através de uma única equação (equação [2.8]), que tem uma característica de equação de ***habilitação*** do evento alocação, isto é:

"Se a operação anterior da rota foi alocada, ***então*** *a operação seguinte poderá ser alocada num slot (instante de tempo) posterior ao slot de término da operação anterior."*

No caso da representação do tempo contínua, não existe a noção de antecedência temporal, tal como na representação uniforme, de modo que esta condição somente poderá ser introduzida através de um conjunto de condições que serão apresentadas a seguir.

3.5 - Formulação Proposta

A formulação alternativa proposta neste trabalho é baseada nos seguintes conceitos:

- Seja S_{il} uma variável positiva e contínua, que indica a existência ou não de estoque da tarefa/produto no estágio l de forma que, se a operação anterior no estágio l-1 tiver sido executada, então a operação relativa ao estágio l do produto/operação i poderá ser realizada. Esta será uma equação semelhante à equação [2.8], que habilita a operação seguinte apenas se anterior tiver sido realizada, então:

$$S_{il} = \sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl-1} - \sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl} \qquad \forall i, l \in L_i \qquad [3.4]$$

Já que $S_{il} \geq 0$, os valores possíveis para esta variável em [3.4] serão:

Tabela 3.7 - Valores possíveis para S_{il}

S_{il}	$\sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl-1}$	$\sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl}$
0	1	1
0	0	0
1	1	0

S_{il} não pode assumir valores negativos, então $\sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl} = 1$ se e somente se $\sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl-1} = 1$.

Note-se que o somatório em **k**, tem como objetivo identificar se a operação foi realizada em algum *slot*, mas não existe qualquer relação de precedência temporal entre *slots* de processadores/equipamentos diferentes.

- Seja H_{il} uma variável positiva e contínua, que indica o instante de tempo em que a operação anterior da rota esteja terminada, e que portanto a operação seguinte pode ser iniciada: (H_{il} representa o instante a partir do qual há estoque)

$$H_{il} = TS_{il-1} + \sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl-1} \; TP_{ij} \qquad \forall i,l \in L_i \qquad [3.5]$$

Assim, o instante de início TS_{il} será:

$$TS_{il} = H_{il} \sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl} \qquad \forall i,l \in L_i \qquad [3.6]$$

A relação [3.6] é não linear, e a forma de linearização a ser empregada é apresentada em Pinto(1995) e Raman et al.(1994):

$$TS_{il} = \tau_{il} \qquad \forall i,l \in L_i \qquad [3.7]$$

$$\tau_{il} \leq U^{il} \sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl} \qquad \forall i,l \in L_i \qquad [3.8]$$

onde:

$\tau_{il} = H_{il} W_{ijkl}$

U^{il} é uma constante suficientemente grande para que não sejam eliminadas soluções factíveis para o problema. As linearizações através do recurso a constantes grandes, em geral, comprometem o cálculo de limitantes inferior durante o processo de solução, já que o problema é mais relaxado. No entanto, em uma estratégia de horizonte rolante este problema é reduzido, já que U^{il}, neste caso, estará associado ao tamanho do horizonte rolante.

A formulação completa a ser utilizada em uma estratégia de horizonte rolante está apresentada na Tabela 3.8 .

Tabela 3.8- Formulação alternativa baseada na representação contínua do tempo para aplicação em uma estratégia de horizonte rolante

$\sum_{j \in (ji \cap jl)} \sum_{k \in Kj} W_{ijkl} \leq 1$	$\forall i,l \in L_i$	[3.3]
$\sum_{i \in Ij} \sum_{l \in (Li \cap Lj)} W_{ijkl} + Y_{jk} = 1$	$\forall j,k \in K_j$	[2.14]
$TES_{jk} \leq TSS_{jk+1}$	$\forall j,k \in K_j - \{k_j\}$	[2.17]
$S_{il} = \sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl-1} - \sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl}$	$\forall i,l \in L_i$	[3.4]
$TE_{il} - DD_{il} + Z_{il} - X_{il} = 0$	$\forall i,l \in L_i$	[3.1]
$Y_{jk} \leq Y_{jk+1}$	$\forall j,k \in K_j - \{k_j\}$	[2.19]
$TS_{il} - TSS_{jk} \geq -U^{il}(1 - W_{ijkl})$	$\forall i,j \in (j_i \cap j_l), k \in K_j, l \in L_i$	[2.20]
$TS_{il} - TSS_{jk} \leq U^{il} W_{ijkl}$	$\forall i,j \in (j_i \cap j_l), k \in K_j, l \in L_i$	[2.21]
$TES_{jk} = TSS_{jk} + \sum_{i \in Ij} \sum_{l \in Li} W_{ijkl}(TP_{ij} + SU_j)$	$\forall j,k \in K_j$	[2.18]
$TE_{il} = TS_{il} + \sum_{j \in (ji \cap jl)} \sum_{k \in Kj} W_{ijkl}(TP_{ij} + SU_j)$	$\forall i,l \in L_i$	[2.16]
$H_{il} = TS_{il-1} - \sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl-1} TP_{ij}$	$\forall i,l \in L_i$	[3.5]
$TS_{il} = \tau_{il}$	$\forall i,l \in L_i$	[3.7]
$\tau_{il} \leq U^{il} \sum_{k \in Kj} \sum_{j \in Ji} W_{ijkl}$	$\forall i,l \in L_i$	[3.8]

A seguir serão apresentados exemplos de aplicações baseados nos dados do exemplo apresentado nas seções 2.7.4 e 3.2.1 .

3.6 - Exemplos de Aplicação

Nestes exemplos serão mostrados a aplicabilidade da formulação proposta para a alocação parcial de operações em um horizonte de planejamento. Os dados utilizados são os apresentados nas seções 2.7.3 e 3.2.1

Serão testadas diferentes funções de custo para os dois exemplos:

- Maximização do número de operações alocadas no horizonte, dada por:

$$Custo = min \sum_{i \in Ij} \sum_{l \in (Li \cap Lj)} \sum_{j \in (ji \cap jl)} \sum_{k \in Kj} (1 - W_{ijkl}) \qquad [3.9]$$

- Minimização do atraso das tarefas, dado por:

$$Custo = min\ Earl = \sum_{i} \sum_{l} X_{il} + \sum_{i} \sum_{l} Z_{il} \qquad [3.2]$$

Exemplo 3.2 - Horizonte de alocação HF=30 para os dados do problema apresentado na seção 2.7.3, sem prazos de entrega dos produtos, minimizando a função de custo [3.9]. Neste caso foi possível a alocação de todas as operações, já que o *makespan* , que é uma medida do tempo total necessário para executar todas as operações, é igual a 24 (Tabela 2.5).

Exemplo 3.3 - Horizonte de alocação HF=15 para os dados do problema apresentado na seção 2.7.3, sem prazos de entrega dos produtos, minimizando a

função de custo [3.9]. Neste caso o custo mínimo atingido foi igual a 2, significando que duas operações deixaram de ser realizadas.

Exemplo 3.4 - Minimização do *earliness*, utilizando-se a equação [3.2] e HF = 30. Os dados do problema foram retirados da seção 3.2.1, mudando-se somente o prazo de entrega da tarefa **A** de 25 para 20. A função de custo obtida foi igual a zero, significando que não houve atraso e nem adiantamento das tarefas, ou seja, os tempos de processamento destas foram iguais aos seus prazos de entrega. a carta de *Gantt* está ilustrada na figura 3.4) .

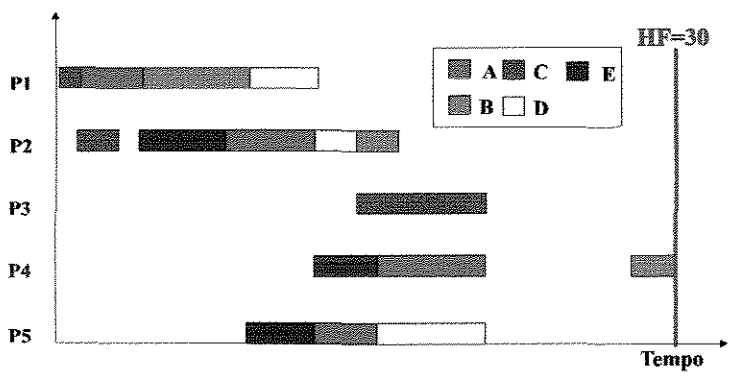


Figura 3.4: Carta de *Gantt* do exemplo 3.4

Exemplo 3.5 - Minimização do *earliness*, utilizando-se a equação [3.2] e HF = 30. Os dados do problema foram retirados da seção 3.2.1, mudando-se somente o prazo de entrega da tarefa **A** de 25 para 10. Obteve-se uma função de custo igual a 8. Neste caso não houve adiantamento de nenhuma tarefa, sendo que as tarefas **A** e **E** sofreram um atraso de 5 e 3 nos seus tempos de processamento, respectivamente. E as demais tarefas (B, C e D) tiveram seus tempos de processamento iguais aos seus prazos de entrega (figura 3.5).

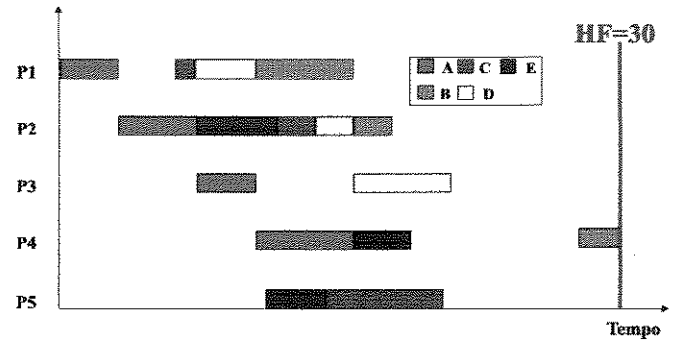


Figura 3.5: Carta de *Gantt* do exemplo 3.5

Exemplo 3.6 - Neste caso, utilizou-se somente quatro processadores, não havendo assim processadores em paralelo. Não foi também estipulados prazos de entrega para as tarefas. O horizonte utilizado foi igual a 20 (HF=20), minimizando a função de custo [3.9]. Obteve-se uma função de custo igual a zero, ou seja, todas as tarefas foram alocadas. A carta de Gantt está ilustrada na figura 3.6 .

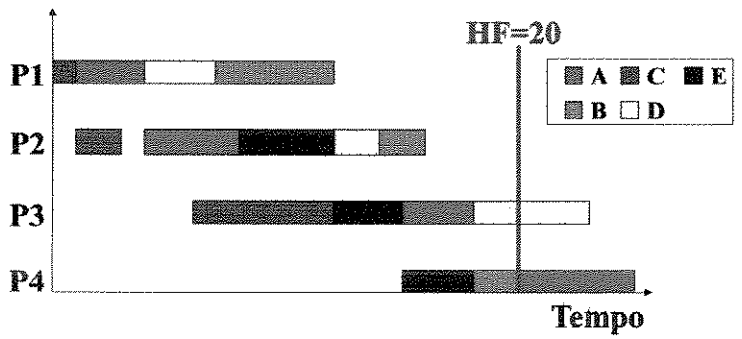


Figura 3.6: Carta de *Gantt* do exemplo 3.6

Exemplo 3.7 - Usou-se as mesmas condições do exemplo 3.6 , sendo o horizonte de 15 (HF=15). Ao se utilizar o OPTCR de 0.3, obteve-se um custo igual a 3, significando que 3 operações não foram alocadas, sendo elas C1, C2 e C3 (figura 3.7a). Porém, utilizando-se um OPTCR de 0.1, todas as operações foram alocadas, obtendo-se um custo igual a zero (figura 3.7b) .

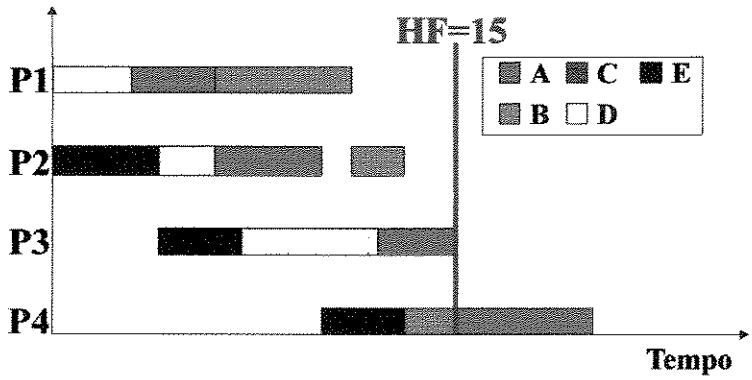


Figura 3.7a: Carta de *Gantt* do exemplo 3.7

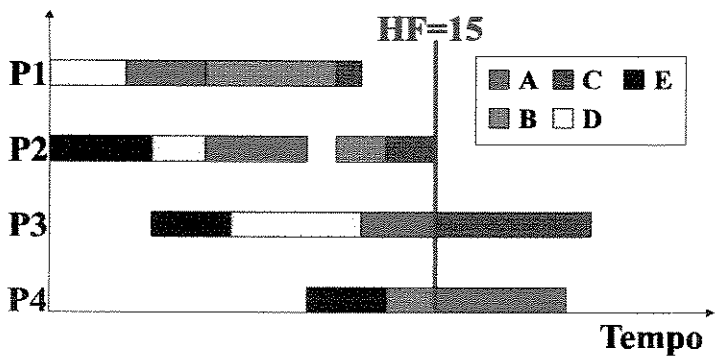


Figura 3.7b: Carta de *Gantt* do exemplo 3.7

Exemplo 3.8 - Utilizou-se também, neste caso, as mesmas condições do exemplo 3.6, porém com um horizonte de 10 (HF=10). Seis operações não foram alocadas (A4,B2,B4,C2,C3,D3), obtendo-se assim um custo igual a 6 (figura 3.8) .

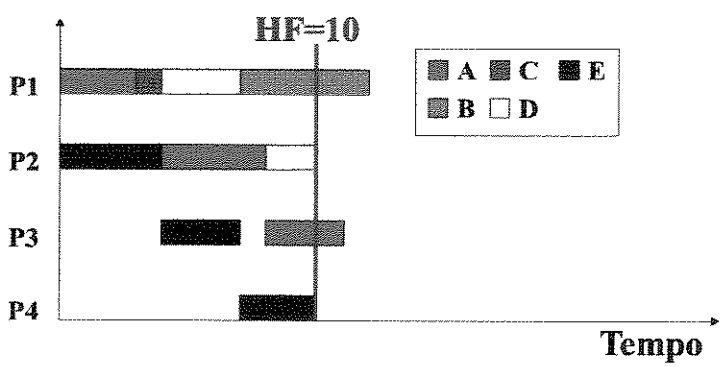


Figura 3.8 Carta de *Gantt* do exemplo 3.8

Exemplo 3.9 - Neste caso, utilizou-se 5 processadores, sendo que dois deles estão em paralelo, como mostra a figura 3.1. Não há prazos de entrega, e a função de custo utilizada é a [3.9]. Com um HF=10, 5 operações não foram alocadas (A4,B2,B4,C2,C3), ou seja, o custo foi igual a 5 (figura 3.9a). Ao se mudar o HF de 10 para 15, todas as operações forma alocadas, obtendo-se um custo igual a zero (figura 3.9b) .

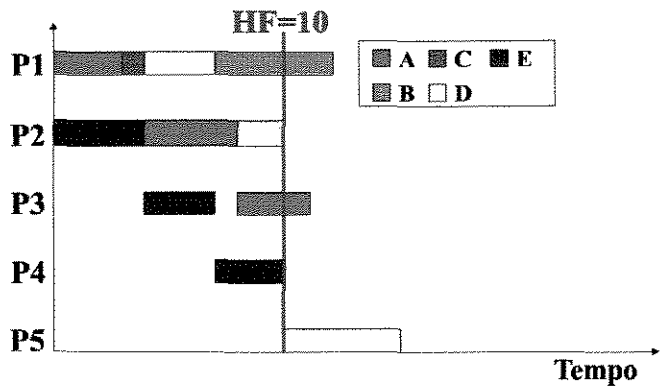


Figura 3.9a: Carta de *Gantt* do exemplo 3.9

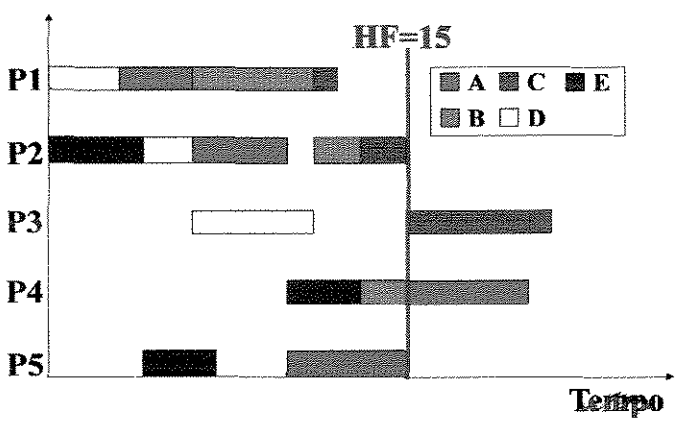


Figura 3.9b: Carta de *Gantt* do exemplo 3.9

Capítulo 4 - Conclusão

Neste trabalho foi proposta uma alteração na formulação baseada em uma Representação Contínua do Tempo, de forma a adaptá-la à estratégia de Horizonte Rolante .

Devido a imprevistos ocorridos em problemas reais, tais como quebra de equipamentos, pode ocorrer dos prazos de entrega não serem cumpridos. Portanto, nesta estratégia é permitido o relaxamento dos prazos de entrega mediante ao pagamento de um custo acrescido na função objetivo .

Sabe-se que os problemas de *scheduling* são classificados como NP-completos e de difícil solução. Porém, ao introduzirmos o conceito de horizonte rolante, o problema é resolvido por partes, diminuindo com isso sua dimensão, dando maior rapidez de resposta a sua solução; no entanto esta é uma estratégia sub-ótima .

A estratégia de horizonte rolante já foi adaptada à formulação baseada em uma discretização uniforme do tempo (Latre et al.,1996), na qual foi capaz de resolver grandes problemas, obtendo soluções satisfatórias e animadoras .

Até o momento não existe uma abordagem geral capaz de resolver qualquer tipo de problema de *scheduling*. Por isso, antes de desenvolver uma estratégia para adaptá-la ao problema em questão, é necessário um estudo detalhado do mesmo. Na literatura existem diversas formulações para solucionar os problemas de programação da produção; no entanto, grande parte destas são para problemas de pequeno porte, e cada uma para um tipo específico de problema. Por isso, antes de compararmos formulações, é necessário verificar para que tipo de problema ela foi

desenvolvida, evitando com isso uma condenação injusta, mesmo porque todas as formulações possuem suas limitações .

Os resultados obtidos no capítulo 3 mostram a grande vantagem de se trabalhar com prazos de entrega relaxados. Pois neste caso, mesmo que algum prazo não possa ser cumprido, o problema encontrará uma solução, aproximando desse modo aos problemas reais. Além disso, ao trabalharmos com um problema de horizonte finito, pode ocorrer de algumas tarefas não serem alocadas; entretanto, estas sofrerão uma penalidade .

Como proposta para trabalhos futuros pode-se citar :

- Adaptação completa à estratégia de Horizonte Rolante a formulação com Representação Contínua do Tempo. Devendo ser explorado com recursos compartilhados, já que estes aumentam, neste tipo de formulação, o número de variáveis binárias, tornando um problema de difícil solução .

Apêndice 1

```
$TITLE JM1/Domínio de tempo continuo/Sem recursos
$OFFUPPER
$OFFSYMXREF
$OFFSYMLIST

SETS TAR Tarefas/T1*T4/
     PROC Processadores/P1*P3/
     SLOT Slots/S1*S4/;

ALIAS(TAR,TARL);
ALIAS(PROC,PROCL);

SETS TAR_PROC(TAR,PROC) Tarefas executadas nos processadores
     /T1. (P1,P2,P3)
      T2. (P1,P2,P3)
      T3. (P1,P2,P3)
      T4. (P1,P2,P3)/;

SETS ULT_SLOT(SLOT);
     ULT_SLOT(SLOT)=YES;
     ULT_SLOT('S4')=NO;

SETS ULT_PROC(PROC);
     ULT_PROC(PROC)=YES;
     ULT_PROC('P3')=NO;

TABLE TP(TAR,PROC) Tempo de processamento
      P1        P2      P3
T1     4         5       2
T2     3         4       3
T3     3         4       6
T4     4         3       6;

SCALAR N Numero de tarefas
       M Numero de processadores
       SS Numero de slots
       U Dicotomia de ordenamento;
       N=CARD(TAR);
       M=CARD(PROC);
       SS=CARD(SLOT);
       U=70;

VARIABLES W(TAR,PROC,SLOT) Tarefa TAR em PROC e SLOT
          Y(PROC,SLOT) Variavel de folga
          TSS(PROC,SLOT) Tempo de inicio de TAR em SLOT
          TES(PROC,SLOT) Tempo de fim de TAR em SLOT
          TS(TAR,PROC) Tempo de inicio de TAR em PROC
          TE(TAR,PROC) Tempo de fim de TAR em PROC
          MSPAN Makespan;

POSITIVE VARIABLES TSS,TES,TS,TE;
BINARY VARIABLES W,Y;

EQUATIONS CUSTO Makespan
          ASS1(TAR,PROC)
          ASS2(PROC,SLOT)
          TIMS1(PROC,SLOT)
          TIMS2(PROC,SLOT)
          TIM1(TAR,PROC)
          TIM2(TAR,PROC)
```

```
          CLIPP1(TAR,PROC,SLOT)
          CLIPP2(TAR,PROC,SLOT);

CUSTO..MSPAN=E=TES('P3','S4');

ASS1(TAR,PROC) $(TAR_PROC(TAR,PROC))..SUM(SLOT,W(TAR,PROC,SLOT))=E=1;

ASS2(PROC,SLOT)..SUM(TAR $(TAR_PROC(TAR,PROC)),W(TAR,PROC,SLOT))+
                 Y(PROC,SLOT)=E=1;

TIMS1(PROC,SLOT)..TES(PROC,SLOT)=E=TSS(PROC,SLOT)+
                  SUM(TAR $(TAR_PROC(TAR,PROC)),
                  W(TAR,PROC,SLOT)*TP(TAR,PROC));

TIMS2(PROC,SLOT) $(ULT_SLOT(SLOT))..TES(PROC,SLOT)=L=
                 TSS(PROC,SLOT+1);

TIM1(TAR,PROC) $(TAR_PROC(TAR,PROC))..
               TE(TAR,PROC)=E=TS(TAR,PROC)+TP(TAR,PROC);

TIM2(TAR,PROC) $(TAR_PROC(TAR,PROC) * ULT_PROC(PROC))..
               TE(TAR,PROC)=L=TS(TAR,PROC+1);

CLIPP1(TAR,PROC,SLOT) $(TAR_PROC(TAR,PROC))..
       TS(TAR,PROC)-TSS(PROC,SLOT)=G=-U*(1-W(TAR,PROC,SLOT));

CLIPP2(TAR,PROC,SLOT) $(TAR_PROC(TAR,PROC))..
       TS(TAR,PROC)-TSS(PROC,SLOT)=L=U*(1-W(TAR,PROC,SLOT));

MODEL JM/ALL/;
OPTION LIMROW=0;
OPTION LIMCOL=0;
OPTION SOLPRINT=OFF;
OPTION OPTCR=0.01;
OPTION ITERLIM=100000;
OPTION RESLIM=10000;
SOLVE JM USING MIP MINIMIZING MSPAN;
DISPLAY
W.L,TSS.L,TES.L,TS.L,TE.L,JM.NODUSD,JM.ITERUSD,JM.RESUSD,MSPAN.L;
```

Apêndice 2

```
$TITLE JM2/Dominio de tempo continuo/com recursos
$OFFUPPER
$OFFSYMXREF
$OFFSYMLIST

SETS TAR Tarefas/T1*T4/
     PROC Processadores/P1*P3/
     SLOT Slots/S1*S4/;

ALIAS(TAR,TARL);
ALIAS(PROC,PROCL);

SETS TAR_PROC(TAR,PROC) Tarefas executadas nos processadores
     /T1.          (P1,P2,P3)
      T2.          (P1,P2,P3)
      T3.          (P1,P2,P3)
      T4.          (P1,P2,P3)/;

SETS ULT_SLOT(SLOT);
     ULT_SLOT(SLOT)=YES;
     ULT_SLOT('S4')=NO;

SETS ULT_PROC(PROC);
     ULT_PROC(PROC)=YES;
     ULT_PROC('P3')=NO;

SETS TAR_TARL(TAR,TARL);
     TAR_TARL(TAR,TARL)=NO;
     LOOP(TAR,
     LOOP(TARL,
     IF(ORD(TAR) NE ORD(TARL),
     TAR_TARL(TAR,TARL)=YES;
     );););

SETS PROC_PROCL(PROC,PROCL);
     PROC_PROCL(PROC,PROCL)=NO;
     LOOP(PROC,
     LOOP(PROCL,
     IF(ORD(PROC) NE ORD(PROCL),
     PROC_PROCL(PROC,PROCL)=YES;
     );););

TABLE TP(TAR,PROC) Tempo de processamento
       P1         P2       P3
T1      4          5        2
T2      3          4        3
T3      3          4        6
T4      4          3        6;

TABLE CR(TAR,PROC) Consumo de recursos
       P1         P2       P3
T1      8          6        4
T2      6          2        4
T3      7          3        6
T4      3          4        8;

SCALAR N Numero de tarefas
       M Numero de processadores
       SS Numero de slots
       U Dicotomia de ordenamento
```

```
        OF Oferta de recursos;
        N=CARD(TAR);
        M=CARD(PROC);
        SS=CARD(SLOT);
        U=70;
        OF=14;

VARIABLES W(TAR,PROC,SLOT) Tarefa TAR em PROC e SLOT
          Y(PROC,SLOT) Variavel de folga
          TSS(PROC,SLOT) Tempo de inicio de TAR em SLOT
          TES(PROC,SLOT) Tempo de fim de TAR em SLOT
          TS(TAR,PROC) Tempo de inicio de TAR em PROC
          TE(TAR,PROC) Tempo de fim de TAR em PROC
          DS(TAR,PROC,TARL,PROCL)
          DSL(TAR,PROC,TARL,PROCL)
          ZB(TAR,PROC,TARL,PROCL)
          CON(TARL,PROCL)
          MSPAN Makespan;

POSITIVE VARIABLES TSS,TES,TS,TE,ZB,CON;
BINARY VARIABLES W,Y,DS,DSL;

EQUATIONS CUSTO Makespan
          ASS1(TAR,PROC)
          ASS2(PROC,SLOT)
          TIMS1(PROC,SLOT)
          TIMS2(PROC,SLOT)
          TIM1(TAR,PROC)
          TIM2(TAR,PROC)
          CLIPP1(TAR,PROC,SLOT)
          CLIPP2(TAR,PROC,SLOT)
          DS1(TAR,PROC,TARL,PROCL)
          DS2(TAR,PROC,TARL,PROCL)
          DSL1(TAR,PROC,TARL,PROCL)
          DSL2(TAR,PROC,TARL,PROCL)
          ZB1(TAR,PROC,TARL,PROCL)
          ZB2(TAR,PROC,TARL,PROCL)
          ZB3(TAR,PROC,TARL,PROCL)
          CONS(TARL,PROCL)
          RESCONS(TARL,PROCL);

CUSTO..MSPAN=E=TES('P3','S4');

ASS1(TAR,PROC) $(TAR_PROC(TAR,PROC))..SUM(SLOT,W(TAR,PROC,SLOT))=E=1;

ASS2(PROC,SLOT)..SUM(TAR $(TAR_PROC(TAR,PROC)),W(TAR,PROC,SLOT))+
                 Y(PROC,SLOT)=E=1;

TIMS1(PROC,SLOT)..TES(PROC,SLOT)=E=TSS(PROC,SLOT)+
                  SUM(TAR $(TAR_PROC(TAR,PROC)),
                  W(TAR,PROC,SLOT)*TP(TAR,PROC));

TIMS2(PROC,SLOT) $(ULT_SLOT(SLOT))..TES(PROC,SLOT)=L=
                 TSS(PROC,SLOT+1);

TIM1(TAR,PROC) $(TAR_PROC(TAR,PROC))..TE(TAR,PROC)=E=
               TS(TAR,PROC)+TP(TAR,PROC);

TIM2(TAR,PROC) $(TAR_PROC(TAR,PROC) * ULT_PROC(PROC))..
               TE(TAR,PROC)=L=TS(TAR,PROC+1);

CLIPP1(TAR,PROC,SLOT) $(TAR_PROC(TAR,PROC))..
       TS(TAR,PROC)-TSS(PROC,SLOT)=G=-U*(1-W(TAR,PROC,SLOT));
```

```
CLIPP2(TAR,PROC,SLOT) $(TAR_PROC(TAR,PROC))..
       TS(TAR,PROC)-TSS(PROC,SLOT)=L=U*(1-W(TAR,PROC,SLOT));

DS1(TAR,PROC,TARL,PROCL) $(TAR_PROC(TAR,PROC) * TAR_TARL(TAR,TARL) *
                         PROC_PROCL(PROC,PROCL))..
                         TS(TARL,PROCL)-TS(TAR,PROC)=G=
                         -U*(1-DS(TAR,PROC,TARL,PROCL));

DS2(TAR,PROC,TARL,PROCL) $(TAR_PROC(TAR,PROC) * TAR_TARL(TAR,TARL) *
                         PROC_PROCL(PROC,PROCL))..
                         TS(TARL,PROCL)-TS(TAR,PROC)=L=
                         U*DS(TAR,PROC,TARL,PROCL);

DSL1(TAR,PROC,TARL,PROCL) $(TAR_PROC(TAR,PROC) * TAR_TARL(TAR,TARL) *
                          PROC_PROCL(PROC,PROCL))..
                          TS(TARL,PROCL)-TE(TAR,PROC)=G=
                          -U*(1-DSL(TAR,PROC,TARL,PROCL));

DSL2(TAR,PROC,TARL,PROCL) $(TAR_PROC(TAR,PROC) * TAR_TARL(TAR,TARL) *
                          PROC_PROCL(PROC,PROCL))..
                          TS(TARL,PROCL)-TE(TAR,PROC)=L=
                          U*DSL(TAR,PROC,TARL,PROCL);

ZB1(TAR,PROC,TARL,PROCL) $(TAR_PROC(TAR,PROC) * TAR_TARL(TAR,TARL) *
                         PROC_PROCL(PROC,PROCL))..
                         ZB(TAR,PROC,TARL,PROCL)=G=
                         DS(TAR,PROC,TARL,PROCL)-
                         DSL(TAR,PROC,TARL,PROCL);

ZB2(TAR,PROC,TARL,PROCL) $(TAR_PROC(TAR,PROC) * TAR_TARL(TAR,TARL) *
                         PROC_PROCL(PROC,PROCL))..
                         ZB(TAR,PROC,TARL,PROCL)=L=
                         1-DSL(TAR,PROC,TARL,PROCL);

ZB3(TAR,PROC,TARL,PROCL) $(TAR_PROC(TAR,PROC) * TAR_TARL(TAR,TARL) *
                         PROC_PROCL(PROC,PROCL))..
                         ZB(TAR,PROC,TARL,PROCL)=L=
                         DS(TAR,PROC,TARL,PROCL);

CONS(TARL,PROCL) $(TAR_PROC(TARL,PROCL))..CON(TARL,PROCL)=E=
                 SUM((TAR,PROC) $(TAR_PROC(TAR,PROC)),
                 CR(TAR,PROC)*ZB(TAR,PROC,TARL,PROCL));

RESCONS(TARL,PROCL) $(TAR_PROC(TARL,PROCL))..
                    CON(TARL,PROCL)=L=OF;

MODEL JM2/ALL/;

OPTION LIMROW=10;
OPTION LIMCOL=0;
OPTION SOLPRINT=OFF;
OPTION OPTCR=0.01;
OPTION ITERLIM=100000;
OPTION RESLIM=10000;
SOLVE JM2 USING MIP MINIMIZING MSPAN;
DISPLAY W.L,Y.L,TSS.L,TES.L,TS.L,TE.L,CON.L,JM2.NODUSD,JM2.ITERUSD,
        JM2.RESUSD,MSPAN.L;
```

Apêndice 3

```
$TITLE KOND2/Discretizacao uniforme do tempo/Sem recursos
$OFFUPPER
$OFFSYMXREF
$OFFSYMLIST

SETS TAR Tarefas/T1*T4/
     PROC Processadores/P1*P3/
     SLOT/S1*S30/;

ALIAS(TAR,TARL);
ALIAS(PROC,PROCL);
ALIAS(SLOT,SLOTL);

TABLE TP(TAR,PROC) Tempos de Processamento
      P1         P2       P3
T1     4          5        2
T2     3          4        3
T3     3          4        6
T4     4          3        6;

SCALAR N Numero de tarefas
       M Numero de processadores;
       N=CARD(TAR);
       M=CARD(PROC);

VARIABLES W(TAR,PROC,SLOT) Tarefa TAR em PROC em SLOT
          S(TAR,PROC,SLOT) Estoque depois de TAR em SLOT
          MSPAN Makespan;

POSITIVE VARIABLES S;
BINARY VARIABLES W;

EQUATIONS CUSTO(TAR) Makespan
          SOMW(TAR,PROC)
          RES(PROC,SLOT) Restricao de Shah
          INIC(TAR)
          ESTOQUE1(TAR,SLOT)
          ESTOQUE2(TAR,PROC,SLOT);

SOMW(TAR,PROC)..SUM(SLOT,W(TAR,PROC,SLOT))=E=1;

RES(PROC,SLOT)..SUM(TAR,SUM(SLOTL $(ORD(SLOTL) LE ORD(SLOT) AND
                ORD(SLOTL) GE ORD(SLOT)-TP(TAR,PROC)+1),
                W(TAR,PROC,SLOTL)))=E=1;

INIC(TAR)..S(TAR,'P1','S1')=E=1-W(TAR,'P1','S1');

ESTOQUE1(TAR,SLOT) $(ORD(SLOT) GT 1)..
        S(TAR,'P1',SLOT)=E=S(TAR,'P1',SLOT-1)-W(TAR,'P1',SLOT);

ESTOQUE2(TAR,PROC,SLOT) $(ORD(PROC) GE 2)..
        S(TAR,PROC,SLOT)=E=S(TAR,PROC,SLOT-1)+
        (W(TAR,PROC-1,SLOT-(TP(TAR,PROC-1)))-W(TAR,PROC,SLOT));

CUSTO(TAR)..MSPAN=G=SUM(SLOT,W(TAR,'P3',SLOT)*ORD(SLOT))+
            TP(TAR,'P3')-1;
```

```
MODEL KOND/ALL/;
OPTION LIMROW=0;
OPTION LIMCOL=0;
OPTION SOLPRINT=OFF;
OPTION OPTCR=0.01;
OPTION ITERLIM=50000;
OPTION RESLIM=5000;
SOLVE KOND USING MIP MINIMIZING MSPAN;
DISPLAY W.L,S.L,KOND.NODUSD,KOND.RESUSD,KOND.ITERUSD,MSPAN.L;
```

Apêndice 4

```
$TITLE KOND2/Discretizacao uniforme do tempo/Com recursos
$OFFUPPER
$OFFSYMXREF
$OFFSYMLIST

SETS TAR Tarefas/T1*T4/
     PROC Processadores/P1*P3/
     SLOT/S1*S30/;

ALIAS(TAR,TARL);
ALIAS(PROC,PROCL);
ALIAS(SLOT,SLOTL);

TABLE TP(TAR,PROC) Tempos de Processamento
      P1         P2       P3
T1     4          5        2
T2     3          4        3
T3     3          4        6
T4     4          3        6;

TABLE CR(TAR,PROC) Consumo de recuros
      P1         P2       P3
T1     8          6        4
T2     6          2        4
T3     7          3        6
T4     3          4        8;

SCALAR N Numero de tarefas
       M Numero de processadores
       OF Oferta;
       N=CARD(TAR);
       M=CARD(PROC);
       OF=14

VARIABLES W(TAR,PROC,SLOT) Tarefa TAR em PROC em SLOT
          S(TAR,PROC,SLOT) Estoque depois de TAR em SLOT
          MSPAN Makespan
          CON(SLOT);

POSITIVE VARIABLES S,CON;
BINARY VARIABLES W;
EQUATIONS CUSTO(TAR) Makespan
          SOMW(TAR,PROC)
          RES(PROC,SLOT) Restricao de Shah
          INIC(TAR)
          ESTOQUE1(TAR,SLOT)
          ESTOQUE2(TAR,PROC,SLOT)
          CONS(SLOT)
          RESCONS(SLOT);

SOMW(TAR,PROC)..SUM(SLOT,W(TAR,PROC,SLOT))=E=1;

RES(PROC,SLOT)..SUM(TAR,SUM(SLOTL $(ORD(SLOTL) LE ORD(SLOT) AND
   ORD(SLOTL) GE ORD(SLOT)-TP(TAR,PROC)+1),
   W(TAR,PROC,SLOTL)))=E=1;

INIC(TAR)..S(TAR,'P1','S1')=E=1-W(TAR,'P1','S1');

ESTOQUE1(TAR,SLOT) $(ORD(SLOT) GT 1)..
        S(TAR,'P1',SLOT)=E=S(TAR,'P1',SLOT-1)-W(TAR,'P1',SLOT);
```

```
ESTOQUE2(TAR,PROC,SLOT) $(ORD(PROC) GE 2)..
        S(TAR,PROC,SLOT)=E=S(TAR,PROC,SLOT-1)+
        (W(TAR,PROC-1,SLOT-(TP(TAR,PROC-1)))-W(TAR,PROC,SLOT));

CUSTO(TAR)..MSPAN=G=SUM(SLOT,W(TAR,'P3',SLOT)*ORD(SLOT))+
             TP(TAR,'P3')-1;

CONS(SLOT)..CON(SLOT)=E=SUM(TAR,SUM(PROC,SUM(SLOTL
             $(ORD(SLOTL) LE ORD(SLOT) AND ORD(SLOTL) GE ORD(SLOT)-
             TP(TAR,PROC)+1),CR(TAR,PROC)*W(TAR,PROC,SLOTL))));

RESCONS(SLOT)..CON(SLOT)=L=OF;

MODEL KOND2/ALL/;
KOND2.WORKSPACE=10;

OPTION LIMROW=0;
OPTION LIMCOL=0;
OPTION SOLPRINT=OFF;
OPTION OPTCR=0.01;
OPTION ITERLIM=100000;
OPTION RESLIM=10000;
SOLVE KOND2 USING MIP MINIMIZING MSPAN;
DISPLAY W.L,S.L,CON.L,KOND2.NODUSD,KOND2.RESUSD,KOND2.ITERUSD, MSPAN.L;
```

Apêndice 5

```
*************************** JMP_ROT1.INC
*********************************
***************       Planta e Rotas das tarefas
**********************
*************************************************************************
******
*                          ------ planta -----
SETS
    SLOT Slots /S1*S5/
    EST Estagios/L1*L4/

    PROC_EST(PROC,EST) Processadores pertencentes aos estagios
      /P1. (L1)
       P2. (L2)
       P3. (L3)
       P4. (L4)
       P5. (L3)/;

ALIAS (PROC,PROCL);
ALIAS (SLOT,SLOTL);
ALIAS(OPR,OPRL);
ALIAS (EST,ESTL);
ALIAS(TAR,TARL);

*************************************************************************
**
SETS OPER_FIM(OPR) Ultima operacao de cada tarefa
     /A4,B4,C3,D3,E4/;

SETS OPER_IN(OPR) Primeira operacao de cada tarefa
     /A1,B1,C1,D1,E2/;

SETS  ANT_OPRL(OPR,OPRL) OPR antecede OPRL

/A1.A2,A2.A3,A3.A4,B1.B2,B2.B4,C1.C2,C2.C3,D1.D2,D2.D3,E2.E3,E3.E4/;

SETS PROC_OPR(PROC,OPR) Processadores habilitados por operacao
     /P1. (A1,B1,C1,D1)
      P2. (A2,B2,C2,D2,E2)
      P3. (A3,C3,D3,E3)
      P4. (A4,B4,E4)
      P5. (A3,C3,D3,E3)/;

SETS SLOT_PROC(SLOT,PROC) Slots predeterminados para procesador
     /(S1*S5).  P1
      (S1*S5).  P2
      (S1*S5).  P3
      (S1*S5).  P4
      (S1*S5).  P5/;

SETS EST_OPR(EST,OPR) Estagios pertencentes aos processadores
     /L1. (A1,B1,C1,D1)
      L2. (A2,B2,C2,D2,E2)
      L3. (A3,C3,D3,E3)
      L4. (A4,B4,E4)/;

SETS ULT_SLOT(SLOT);
     ULT_SLOT(SLOT)=YES;
     ULT_SLOT('S5')=NO;

SETS EST_ESTL(EST,ESTL);
```

```
      EST_ESTL(EST,ESTL)=NO;
      LOOP(EST,
      LOOP(ESTL,
      IF(ORD(EST) NE ORD(ESTL),
      EST_ESTL(EST,ESTL)=YES;
      ); ); );

SETS  OPR_OPRL(OPR,OPRL);
      OPR_OPRL(OPR,OPRL)=NO;
      LOOP(OPR,
      LOOP(OPRL,
      IF(ORD(OPR) NE ORD(OPRL),
      OPR_OPRL(OPR,OPRL)=YES;
      ); ); );

SETS  ROTA(OPR,EST,OPRL,ESTL);
      ROTA(OPR,EST,OPRL,ESTL)=NO;
      ROTA('A1','L1','A2','L2')=YES;
      ROTA('A2','L2','A3','L3')=YES;
      ROTA('A3','L3','A4','L4')=YES;
      ROTA('B1','L1','B2','L2')=YES;
      ROTA('B2','L2','B4','L4')=YES;
      ROTA('C1','L1','C2','L2')=YES;
      ROTA('C2','L2','C3','L3')=YES;
      ROTA('D1','L1','D2','L2')=YES;
      ROTA('D2','L2','D3','L3')=YES;
      ROTA('E2','L2','E3','L3')=YES;
      ROTA('E3','L3','E4','L4')=YES;

SETS  TAR Tarefas/A,B,C,D,E/;

SETS  OPR Operacoes/A1*A4,B1*B2,B4,C1*C3,D1*D3,E2*E4/;

SETS  PROC Processadores/P1*P5/;

******************************** JMP_TEM1.INC
******************************
*     Tempos de processamento
************************ Tempos de  processamento
************************
TABLE TP(TAR,PROC) Tempos de processamento das tarefas
          P1    P2    P3    P4    P5
A         3     4     3     5     3
B         5     2     0     2     0
C         1     2     6     0     6
D         3     2     5     0     5
E         0     4     3     3     3;


TABLE TPOPR(OPR,PROC) Tempos de processamento das operacoes
      P1     P2     P3     P4     P5
A1    3      0      0      0      0
A2    0      4      0      0      0
A3    0      0      3      0      3
A4    0      0      0      5      0
B1    5      0      0      0      0
B2    0      2      0      0      0
B4    0      0      0      2      0
C1    1      0      0      0      0
C2    0      2      0      0      0
C3    0      0      6      0      6
D1    3      0      0      0      0
D2    0      2      0      0      0
D3    0      0      5      0      5
```

```
E2    0     4     0     0     0
E3    0     0     3     0     3
E4    0     0     0     3     0;

*****************************JMP_DDH1.INC*****************************
*

PARAMETERS DD(TAR) Due date das tarefas
           /A    10
            B    30
            C    20
            D    20
            E    15/;

PARAMETERS DD_OPR(OPR) Due date das operacoes
            /A4   10
             B4   30
             C3   20
             D3   20
             E4   15/;

PARAMETERS EBT(TAR) Inicio da janela
           /A     0
            B     0
            C     0
            D     0
            E     0/;

PARAMETERS INJ(OPR) Inicio da janela
           /A1     0
            B1     0
            C1     0
            D1     0
            E2     0/;

PARAMETERS H(OPR) Pesos para os atrazos
           /A4    1
            B4    1
            C3    1
            D3    1
            E4    1/;


*****************************VARIAB1.INC*****************************
*

VARIABLES W(OPR,PROC,SLOT,EST) Estagio EST designado para OPR em PROC
e SLOT
          Y(PROC,SLOT) Variavel de folga
          TSS(PROC,SLOT) Tempo de inicio de proc em SLOT
          TES(PROC,SLOT) Tempo de fim de PROC em SLOT
          TS(OPR,EST) Tempo de inicio de OPR em EST
          TE(OPR,EST) Tempo de fim de OPR em EST
          Z(OPR,EST)
          X(OPR,EST)
          COST
          COST1
          COST2
          COST3;

POSITIVE VARIABLES TSS,TES,TS,TE,Z,X;
BINARY VARIABLES W,Y;
```

```
*******************************
SCALAR1.INC*******************************

SCALAR N Numero de tarefas
       M Numero de processadores
       SS Numero de slots
       L Numero de estagios
       HF Horizonte
       ALFA
       BETA
       GAMA
       U Dicotomia de ordenamento;
       N=CARD(TAR);
       M=CARD(PROC);
       SS=CARD(SLOT);
       L=CARD(EST);
       HF=30;
       ALFA=100;
       BETA=10;
       GAMA=10;
       U=50;


*********************************MODEL1.INC*****************************
*

EQUATIONS CUSTO
          CUSTO1
          CUSTO2
          CUSTO3
          DDH(OPR,EST)
          ASS1(OPR,EST)
          ASS2(PROC,SLOT)
          ASS3(PROC,SLOT)
          TIMS1(PROC,SLOT)
          TIMS2(PROC,SLOT)
          TIM1(OPR,EST)
          TIM2(OPR,EST)
          TIM3(OPR,EST)
          TIM4(OPR,EST,OPRL,ESTL)
          CLIPP1(OPR,PROC,SLOT,EST)
          CLIPP2(OPR,PROC,SLOT,EST);

$ONTEXT
CUSTO1..COST1=E=(SUM((OPR,EST) $(EST_OPR(EST,OPR)),
       (1-SUM(PROC $(PROC_OPR(PROC,OPR) * PROC_EST(PROC,EST)),
       SUM(SLOT $(SLOT_PROC(SLOT,PROC)),W(OPR,PROC,SLOT,EST))))));
$OFFTEXT

CUSTO2..COST2=E=(SUM((OPR,EST) $(OPER_FIM(OPR) *
EST_OPR(EST,OPR)),X(OPR,EST)));

CUSTO3..COST3=E=(SUM((OPR,EST) $(OPER_FIM(OPR) *
EST_OPR(EST,OPR)),Z(OPR,EST)));

CUSTO..COST=E=BETA*COST2+GAMA*COST3;

DDH(OPR,EST) $(OPER_FIM(OPR) * EST_OPR(EST,OPR))..
                H(OPR)*(TE(OPR,EST)-DD_OPR(OPR)+Z(OPR,EST)-
X(OPR,EST))=E=0;


ASS1(OPR,EST) $(EST_OPR(EST,OPR))..SUM((PROC,SLOT)
$(PROC_EST(PROC,EST) *
```

```
                SLOT_PROC(SLOT,PROC) * PROC_OPR(PROC,OPR)),
                W(OPR,PROC,SLOT,EST))=E=1;


ASS2(PROC,SLOT) $(SLOT_PROC(SLOT,PROC))..SUM((OPR,EST)
                $(EST_OPR(EST,OPR) * PROC_EST(PROC,EST) *
                PROC_OPR(PROC,OPR)),
                W(OPR,PROC,SLOT,EST))+Y(PROC,SLOT)=E=1;

ASS3(PROC,SLOT) $(SLOT_PROC(SLOT,PROC) * ULT_SLOT(SLOT))..
                Y(PROC,SLOT)=L=Y(PROC,SLOT+1);

TIMS1(PROC,SLOT) $(SLOT_PROC(SLOT,PROC))..
                 TES(PROC,SLOT)=E=TSS(PROC,SLOT)+
                 SUM((OPR,EST)
                 $(EST_OPR(EST,OPR) * PROC_EST(PROC,EST) *
                 PROC_OPR(PROC,OPR)),
                 W(OPR,PROC,SLOT,EST)*
                 (TPOPR(OPR,PROC)));

TIMS2(PROC,SLOT) $(SLOT_PROC(SLOT,PROC) * ULT_SLOT(SLOT))..
                 TES(PROC,SLOT)=L=TSS(PROC,SLOT+1);

TIM1(OPR,EST) $(EST_OPR(EST,OPR))..
              TE(OPR,EST)=E=TS(OPR,EST)+
              SUM((PROC,SLOT) $(PROC_EST(PROC,EST) *
              SLOT_PROC(SLOT,PROC) * PROC_OPR(PROC,OPR)),
              W(OPR,PROC,SLOT,EST)*
              (TPOPR(OPR,PROC)));

TIM2(OPR,EST) $(OPER_IN(OPR) * EST_OPR(EST,OPR))..
                   TS(OPR,EST)=G=INJ(OPR);

TIM3(OPR,EST) $(EST_OPR(EST,OPR))..TS(OPR,EST)=L=HF;


TIM4(OPR,EST,OPRL,ESTL) $(ROTA(OPR,EST,OPRL,ESTL) *
EST_ESTL(EST,ESTL) *
                   OPR_OPRL(OPR,OPRL))..
                   TE(OPR,EST)=L=TS(OPRL,ESTL);

CLIPP1(OPR,PROC,SLOT,EST) $(PROC_EST(PROC,EST) *
                          SLOT_PROC(SLOT,PROC) * PROC_OPR(PROC,OPR) *
                          EST_OPR(EST,OPR))..
                          TS(OPR,EST)-TSS(PROC,SLOT)=G=
                           -U*(1-W(OPR,PROC,SLOT,EST));

CLIPP2(OPR,PROC,SLOT,EST) $(PROC_EST(PROC,EST) *
                          SLOT_PROC(SLOT,PROC) * PROC_OPR(PROC,OPR) *
                          EST_OPR(EST,OPR))..
                          TS(OPR,EST)-TSS(PROC,SLOT)=L=
                          U*(1-W(OPR,PROC,SLOT,EST));



$TITLE Jose Mauricio/JM5T1
$OFFUPPER
$OFFSYMXREF
$OFFSYMLIST

$INCLUDE JMP_TEM1.INC
$INCLUDE JMP_ROT1.INC
$INCLUDE JMP_DDH1.INC
```

```
$INCLUDE VARIAB1.INC
$INCLUDE SCALAR1.INC
$INCLUDE MODEL1.INC

MODEL JM5T1/ALL/;
JM5T1.WORKSPACE=10;

OPTION LIMROW=20;
OPTION LIMCOL=0;
OPTION SOLPRINT=OFF;
OPTION SYSOUT=OFF;
OPTION OPTCR=0.5;
OPTION ITERLIM=100000;
OPTION RESLIM=10000;
SOLVE JM5T1 USING MIP MINIMIZING COST;
DISPLAY W.L,Y.L,Z.L,X.L,TSS.L,TES.L,TS.L,TE.L,COST.L,JM5T1.NODUSD,
        JM5T1.RESUSD,JM5T1.ITERUSD;
```

Referências Bibliográficas

Baker, K.R. "Introduction to Sequencing and Scheduling", Jonh Wiley and Sons, New York, 1973

Brooke, A. , Kendrick, D., Meeraus A. "GAMS - A User's Guide", The Scientific Press, Redwood City, USA, 1988

Campos, M.F.D. "Sequenciamento e Alocação de Operações em Flow-shops com Restrições sobre os Recursos Compartilhados e sobre os Prazos de Entrega das Tarefas: Uma Abordagem de Busca orientada por Restrições", Tese de Mestrado, Unicamp, agosto de 1993

Ecker, J.G., Kupferschmid, M. "Introduction to Operation Research", Wie Wiley, 1988

Egli, U.M., Rippin, D.W. "Short-Term Scheduling for Multiproduct Batch Chemical Plants", Computers Chem. Eng., vol. 10, n. 4, pp. 303-325, 1986

Grossmann, I.E., Queseda, I., Raman, R., Voudouris, V.T. "Mixed-Integer Optimization Techniques for the Design and Scheduling of Batch Processes", Department of Chemical Engineering and Design Research Center, Carnegie Mellon University, Pittsburgh, U.S.A., 1992

Karimi, I., Ku, H. "A Modified Heuristic for An Initial Sequence in Flow Shop Scheduling", Ind. Eng. Chem. Res., vol. 17, pp. 1654-1658, 1988

Kondili, E., Pantelides, C.C., Sargent, R.W.H. "A General Algorithm for Short-Term Scheduling of Batch Operations - I. MILP Formulation", Computers Chem. Eng., vol.17, n. 2, pp.211-227, 1993

Ku, H.M., Rajagopalan, D., Karimi, I. "Scheduling in Batch Processes", Chem. Eng. Prog., 83(8), pp. 35-45, 1987

Ku, H.M., Karimi, I. "Scheduling in Serial Multiprodut Batch Processes with Finite Interstage Storange: A Mixed Integer Linear Program Formulation", Ind. Eng. Chem. Res., vol. 27, n. 10, pp. 1840-1848, 1988

Ku, H.M., Karimi, I. "Scheduling in Serial Multiproduct Batch Processes with Due-Date Penalties", Ind. Eng. Chem. Res., vol. 29, n.4, pp. 580-590, 1990

Latre, L.G., Campos, M.D., Rodrigues, M.T.M., Passos, C.A.S., "A Dispatch Rule with Rolling Prediction Horizon for Chemical Flowshops", Conference on Computer Integrated Manufacturing in The Process Industries (CIMPRO'94), pp. 102-117, New Brunswick, USA, 1994

Latre, L.G., Rodrigues, M.T.M., Passos, C.A.S., Campos, M.D. "Multiproduct Batch Plants Scheduling Using a Rolling Horizon and a Lookahead Procedure", Proceeding of the Second International Conference on Computer Integrated Manufacturing in the Process Industries, (CIMPRO'96), pp. 270-284, Eindhoven, Netherlands, 1996

Latre, L.G., Rodrigues, M.T.M., Passos, C.A.S., Campos, M.D. "Reactive Scheduling Approach for Multipurpose Chemical Batch Plants", Comuters Chem. Eng., vol. 20, pp. 1215-1220, ESCAPE, 1996

Lázaro, M., Espuña, A., Puigjaner, L. "A Comprehensive Approach to Production Planning in Multipurpose Batch Plants", Computers Chem. Eng., vol. 13, n. 9, pp. 1031-1047, 1989

Leinstein, R. "Flow-Shop Sequencing Problems with Limited Buffer Storage", Int. J. Prod. Res., vol. 28, n. 11, pp. 085-2100, 1990

Medeiros, A.C.G. "Estratégia de Simulação para a Alocação de Produtos em Plantas Multipropósito", Tese de Mestrado, Unicamp, abril de 1995

Musier, R.F.H., Evans, L.B. "An Approximate Method for the Production Scheduling of Industrial Batch Process with Parallel Units", Computers Chem. Eng., 13(1/2), pp. 229, 1989

Passos, C.A.S., Latre, L.G., Rodrigues, M.T.M., Campos, M.F.D. "Seqüenciamento de Plantas Químicas Multiprodutos: Uma Abordagem Multi Nível", Congresso Brasileiro de Automática, SBA, Rio de Janeiro, 1994

Pinto, J.M. and Grossmann, I.E. "Resource Constrained Model for Short Term Scheduling of Batch Plants", Carnegie Mellon University, Pittsburg, 1994

Pinto, J.M. "Mixed Integer and Logic Based Optimization Techniques for Scheduling of Multistage Chemical Processes", Ph.D. Thesis, Carnegie Mellon University, Pittsburg, Pennsylvania, September, 1995

Reklaitis, G.V. "Review of Scheduling of Process Operation", AIChe Symposium Series, vol. 78, n. 214, pp. 119-133, 1982

Rich, S., Prokopakis, G.J. "Scheduling and Sequencing of Batch Operations in a Multiproduct Plant", Ind. Eng. Chem. Proc. Dev., 25, pp. 979, 1986

Rodrigues, M.T.M. "Seqüenciamento e Alocação de Operações em Flow Shops na Indústria Química com Restrições sobre os Recursos Compartilhados: Uma Abordagem de Busca Orientada por Restrições", Tese de Doutorado, Unicamp, agosto de 1992

Sahinidis, N.V., Grossmann, I.E., Fornari, R.E., Chathrathi, M. "Optimization Model for Long Range Planning in the Chemical Industry", Computers Chem. Eng., vol. 13, n. 9, pp. 1049-1063, 1989

Shah, N., Pantelides, C.C., Sargent, R.W.H., "A General Algorithm for Short-Term Scheduling of Batch Operations-II. Computational Issues", Comp. Chem. Eng., vol. 17, n. 2, pp. 229-244, 1993

ABSTRACT

Recently, much attention has been focused on schenduling problems in the process industry as a consequence of its economic importance. New approaches to solve these problems have been presented during the last 3 years seeking to properly chemical processes, such as recycles.

However, these approaches can't still be applied to industrial problems. In this work a new procedure to deal with industrial problems has been presented. This procedure is based on rolling horizons, such as used in predictive control problems. Through this procedure, the original case can be divided in many smaller cases easier to be studied. Since these smaller cases are obtained, different strategies can be applied to solve them: Branch and Bound, Heuristics and Mathematical Programming.

In this work a continuos time representation was studied and used to solve scheduling problems in the rolling horizons approach. Many different techniques found in the litterature were studied and adapted to solve multipurpose problems.