UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA
DEPARTAMENTO DE ENGENHARIA DA COMPUTAÇÃO E
AUTOMAÇÃO INDUSTRIAL
AGOSTO DE 1992

SEQUENCIAMENTO E ALOCAÇÃO DE OPERAÇÕES EM FLOW SHOPS NA INDÚSTRIA QUÍMICA COM RESTRIÇÕES SOBRE OS RECURSOS COMPARTILHADOS: UMA ABORDAGEM DE BUSCA ORIENTADA POR RESTRIÇÕES

Por :Maria Teresa Moreira Rodrigues

Orientador:Prof Dr. Luis Gimeno Latre

Tese apresentada à Faculdade de Engenharia Elétrica FEE - UNICAMP como parte dos requisitos exigidos para a obtenção do título DOUTOR EM ENGENHARIA

Campinas - 1992

AGRADECIMENTOS

Agradeço especialmente ao Prof. Dr. Luis Gimeno Latre pela orientação e amizade durante os "muitos anos" de duração deste trabalho, e que ele continue.....

Agradeço ao Prof. Dr. Mário de Jesus Mendes da Faculdade de Engenharia Química pelo incentivo, apoio e amizade, e pela minha iniciação na vida acadêmica.

Agradeço a Carlos Alberto Passos, pelas frutíferas discussões às quintas-feiras.

Agradeço aos professores do DCA que durante o tempo de duração deste trabalho me acolheram com amizade.

RESUMO

O problema do scheduling em unidades químicas flexíveis tem crescido em importância nos últimos anos, especialmente na indústria farmacêutica e da química fina.

Em função da natureza físico-química dos produtos processados na indústria química, a estrutura de processamento multi-estágios com fluxo unidirecional, denominada flow shop, é a mais empregada. Estes processos se caracterizam pelo uso intensivo da armazenagem intermediária e outros recursos compartilhados tais como energia elétrica, vapor etc.

Devido à sazonalidade da demanda e/ou à vida curta dos produtos manufaturados nas unidades flexíveis, os planos de produção de curto prazo, denominado scheduling, são freqüentemente alterados, e exigem o recurso a algum tipo de ferramenta capaz de auxiliar na decisão de em qual ordem e quando cada uma das operações necessárias à produção de cada produto devem ser realizadas.

Este tipo de problema é complexo devido à sua natureza combinatória, e em sua forma mais simples denominada sequenciamento, tem sido resolvido recorrendo-se a ferramentas tais como programação matemática, métodos de busca e procedimentos heurísticos. Para o caso de existirem restrições na utilização dos recursos compartilhados, o problema se torna mais complexo.

Neste trabalho optou-se por utilizar um método de busca em árvore do tipo Branch and Bound (BAB) para programar de forma factível a seqüência de produtos minimizando o tempo total de execução das tarefas, chamado makespan.

A eficiência de uma abordagem deste tipo está fortemente associada à qualidade do custo estimado em cada nó da árvore, chamado "lower bound". Quanto melhor a estimativa menor o backtracking.

Neste trabalho foi desenvolvida uma abordagem capaz de calcular o lower bound em cada nó tendo em conta o efeito dos recursos comuns sobre o valor do makespan.

Esta abordagem se baseia em dividir o horizonte do schedule em cada nó da árvore em três regiões. A primeira região é definida entre t=0 e o instante de término da primeira operação da última tarefa da seqüência parcial, e o perfil de demanda de recursos é estático e factível. Na segunda região, definida entre o término da primeira região e o instante de término da última operação da última tarefa da seqüência parcial, existe um perfil parcial de demanda e é analisada a possibilidade de utilizar parcial ou integralmente a oferta excedente de recurso, se ela existir. Na terceira região, definida entre o término da segunda região e o valor estimado do lower bound, admite-se que o volume de recurso será integralmente utilizado.

A abordagem proposta se aproxima das abordagens atuais de scheduling orientado por recursos.

ÍNDICE

CAPÍTULO 1 - INTRODUÇÃO

O problema de Planejamento e Programação da Produção em unidades químicas flexíveis, tem adquirido importância em função de dois aspectos: crescimento do mercado de produtos de química fina e farmacêutica, e necessidade de melhoria do desempenho de unidades químicas tradicionais tipicamente flexíveis. As principais características da indústria de química fina e farmacêutica são a produção em batelada em unidades de pequeno porte, grande número de produtos produzidos na mesma unidade e alto valor agregado dos produtos. Dentre as unidades tradicionais podem se destacar como tipicamente flexíveis a indústria do aço, alimentícia e papel. Embora o espectro de valores agregados, número de produtos, número de equipamentos e outras características, seja bastante amplo, estas indústrias são afetadas pelo mesmo tipo de problema: Quando cada uma das operações de processamento deve ser realizada de forma a cumprir um plano de produção com um custo mínimo. Este é um problema crucial que depende não só de características intrínsecas do processamento dos produtos, tais como tempo de processamento, preparação etc, mas também depende de parâmetros externos à unidade de produção, tais como oferta limitada de recursos externos do tipo energia elétrica, matérias-primas, e de características de demanda do mercado. As características internas, externas e de mercado estabelecem um perfil de produção flexível no tempo, exigindo adaptações freqüentes ou geração de programas de produção.

Esta é uma problemática cuja importância se tornou mais relevante na indústria química na década de 80, coincidindo com a preocupação crescente nas áreas correlatas de simulação e projeto de unidades químicas flexíveis, embora sua importância já tivesse sido ressaltada há muitos anos (McCurdy 1969).

Este trabalho tem dois objetivos. O primeiro é apresentar as principais características das unidades químicas flexíveis de forma a definir os contornos dos principais problemas de Planejamento e Programação da Produção em unidades químicas flexíveis.

Em conseqüência deste estudo foram identificados alguns aspectos importantes:

- Dada a complexidade do problema de Planejamento e Programação da Produção em unidades industriais flexíveis bem como a sua natureza combinatória, ele é normalmente subdividido em no mínimo dois níveis, o Planejamento de Longo Prazo e a Programação de Curto Prazo também chamada de Scheduling.

- Os problemas de scheduling que afetam as unidades flexíveis são comuns às diferentes áreas da atividade industrial. Como consequência as abordagens e ferramentas utilizadas na solução dos problemas são igualmente comuns. Pode-se abstrair portanto da natureza dos produtos, que são normalmente designados por tarefas, e dos equipamentos, normalmente designados por processadores.

- O problema global do scheduling retêm a natureza combinatória característica do problema de Planejamento e Programação da Produção, de maneira que é normalmente classificado como "Hard Problem", por isso a sua solução se dá através de uma abordagem hierárquica, de modo a subdividi-lo

em pequenos problemas manipuláveis, muito embora, dada a interdependência dos diferentes níveis ou subproblemas, esta subdivisão bem como a qualidade da solução final sejam questionáveis. No entanto este tem sido o único caminho capaz de conduzir a construção de soluções implementáveis.

- A abordagem hierárquica consiste em resolver progressivamente os subproblemas do problema original, acrescentando mais informações à medida que o espaço de soluções factíveis ou aceitáveis é reduzido. Deve-se portanto dispor de ferramentas e abordagens eficientes para a solução dos sub-problemas. Dentre os diferentes subproblemas de scheduling importantes especialmente na indústria química destaca-se o o scheduling de tarefas em unidades de processamento multiestágios em batelada com fluxo unidirecional de produtos, denominadas flow shops, com oferta limitada de recursos. Na indústria química estes recursos são normalmente vapor, água, energia elétrica etc. Embora este seja um problema importante, não há até o presente uma abordagem capaz de gerar soluções ótimas. As abordagens normalmente empregadas são adaptações de abordagens para a solução de problemas de sequenciamento, para as quais não se põe o problema de compartilhamento de recursos.

- Tradicionalmente os problemas de scheduling em flow shops são divididos em dois níveis:

Nível 1 - Sequenciamento de tarefas;
Nível 2 - Alocação de recursos;

sendo que geralmente os processadores não são tratados como recursos compartilhados. No caso de haver compartilhamento de recursos, à exceção do processador, os dois níveis não podem mais ser tratados de forma independente.

- As abordagens para tratar o problema de scheduling em sistemas flexíveis com limitação na oferta de recursos comuns são sub-ótimas não só em função do emprego de ferramentas de scheduling sub-ótimas mas também pela dificuldade e o tempo necessário para solucioná-los.

O segundo objetivo deste trabalho é desenvolver e apresentar uma abordagem do tipo Branch and Bound em que o limitante inferior foi especialmente construído para ter em conta o efeito da oferta limitada de recursos sobre a evolução da função objetivo adotada. Ela foi desenvolvida para ser inicialmente utilizada em um sistema hierárquico de planejamento de curto prazo.

A abordagem apresentada certamente não esgota o problema de scheduling em flow shops, mas coloca em evidência algumas dificuldades associadas a este problema, especialmente a interdependência entre os dois níveis citados: seqüenciamento e alocação.

Os algoritmos propostos baseiam-se na idéia que o crescimento da função objetivo ao longo dos ramos a serem sondados é devido a três fatores:

1- A não utilização completa do volume de recursos ofertados no intervalo de tempo comprometido pela seqüência parcial em cada nó da árvore de busca;

2- A parcela de tempo necessária para executar as tarefas remanescentes depende do volume global de recursos ofertados, mas também da oferta instantânea destes mesmos recursos. Se a oferta instantânea for

insuficiente, o perfil de oferta é incapaz de acomodar esta nova demanda, deslocando-se o instante de início da operação da tarefa candidata até que a demanda seja compatível com a oferta .

3- Naquelas condições em que a oferta é capaz de suprir a demanda, o tempo mínimo necessário para executar as tarefas não programadas dependerá apenas do compartilhamento eficiente do recurso processador.

A abordagem proposta pode ser encarada como uma abordagem "míope" na medida que explora a influência do compartilhamento de recursos apenas nas vizinhanças da solução parcial que está sendo explorada no nó. Esta característica levou à discussão da possibilidade de utilização de estratégias calcadas no conceito recente de ordenamento de tarefas orientado por recursos. Este conceito é discutido no Capítulo 3, e no Anexo C é apresentada uma proposta inicial de utilização deste conceito no seqüenciamento de tarefas em flow shops. O objetivo futuro será a pesquisa de alternativas de combinar estas duas abordagens de maneira a abreviar a busca de soluções implementáveis para o scheduling de tarefas em flow shops.

CAPÍTULO 2 - CARACTERÍSTICAS GERAIS DOS PROCESSOS QUÍMICOS FLEXÍVEIS

2.1. INTRODUÇÃO

O objetivo deste capítulo é o de situar o problema de scheduling em flow shops na indústria química no contexto de projeto e operação de plantas químicas batelada ou semi-contínuas.

Este capítulo obedece a seguinte organização:

- Apresentação das principais características das plantas químicas descontínuas;
- Apresentação dos principais problemas associados ao projeto, simulação, operação e gerenciamento da produção de plantas químicas descontínuas.

À exceção do gerenciamento da produção, a abordagem dos assuntos é superficial dado que o objetivo é apenas o de descrever o contexto no qual se situa o problema de scheduling.

2.2. Caracterização das plantas batelada e semi-contínuas

As unidades químicas podem ser definidas como aquelas em que as matérias primas são transformadas em produtos em unidades de processamento, onde são realizadas operações físico-químicas.

As principais características destas unidades são apresentadas na Tabela 2.1.

Tabela 2.1 - Características das Unidades Químicas

- Ocorrência de reações químicas
- Entradas e Saídas Quimicamente diferentes
- Os materiais tem natureza "contínua"
- Fluxo de matéria
- Ocupação simultânea das unidades de processamento
- Conectividade das unidades de processamento

No início da indústria química os processos eram até certo ponto "artesanais", e eram realizados em unidades descontínuas (ou batelada), já que este tipo de processamento permite uma intervenção direta do operador nas operações de controle da unidade.À medida que a demanda por produtos químicos foi crescendo, cresceu também o investimento em estudos e projetos para a transformação das unidades descontínuas, que apresentam normalmente baixa produtividade, em unidades contínuas, que apresentam normalmente alta produtividade. No entanto as unidades contínuas exigem conhecimentos mais

profundos do processo. A alteração das condições de processamento, implica na operação da unidade em regime transiente que, por sua vez, implica em um comprometimento da qualidade do produto neste intervalo de tempo.

Nesta transição dos processos não-contínuos para os processos contínuos, os primeiros foram praticamente abandonados, e passaram a ser considerados processos de "segunda classe". Com o advento de vários produtos nas áreas de química fina e biotecnologia voltaram a ser processos de "primeira classe" e portanto a ter espaço na literatura especializada.

Na Tabela 2.2 são apresentados exemplos de alguns processos realizados tradicionalmente em regimes contínuo e não-contínuo (batelada e/ou semi-contínuo).

Tabela 2.2 - Exemplos de Processos Contínuos e Não-Contínuos

Contínuos (Entradas e Saídas sem interrupção)	Não-Contínuos (Entradas e Saídas intermitentes)
Refino de Petróleo Petroquímica, Fertilizantes, Química de Base (Amonia, Ácido Sulfúrico etc)	Bioquímica, Polímeros, Corantes, Cosméticos, Farmacêutica, Herbicidas, Alimentos, Aço

Os principais aspectos que diferenciam os processos contínuos dos não contínuos são apresentados na Tabela 2.3.

É importante ressaltar que as unidades químicas batelada são em geral, a síntese de equipamentos que operam em regime descontínuo e outros cujo modo de operação é tipicamente semi-contínuo.As diferenças entre estes dois tipos de equipamento estão apresentadas na Tabela 2.4.

Existem também processos e conseqüentemente unidades, operadas exclusivamente em regime semi-contínuo e, tal como os processos exclusivamente batelada, constituem unidades flexíveis e apresentam as mesmas características dos processos batelada. As unidades semi-contínuas também são operadas intermitentemente e são especificadas por taxas de processamento, mas as etapas de processamento são idênticas às das unidades descontínuas.

Tabela 2.3 - Diferenças entre os processos contínuos e não-contínuos

Unidades contínuas	Unidades Batelada ou Semi contínuas
Baixa armazenagem intermediária	Alta armazenagem intermediária
Alta estocagem	Baixa estocagem
Regime estacionário	Regime não-estacionário
Unidades dedicadas(não há alocação de tempo)	Unidades flexíveis(alocação de tempo)
Larga escala de produção	Pequena escala de produção
Bem estudadas	Pouco estudadas
Commodities ou produtos de baixo valor agregado	Especialidades Químicas ou produtos de alto valor agregado

Tabela 2.4.- Diferenças entre as unidades descontínuas e operações semi-contínuas

Unidades Descontínuas	Operações Semi-contínuas
-Entradas e saídas intermitentes	-Operação Intermitente (partidas e paradas)
-Especificadas por volume ou dimensão dos equipamentos	-Especificadas por taxas
-Opera em ciclos	-Taxas de operação variáveis
-Etapas de operação:	-Etapas de operação:
.Transferencia de matéria prima para o equipamento	.Processamento
.Processamento	.Limpeza/preparação (set-up)
.Transferencia do produto	.Período ocioso
.Limpeza e preparação	
Equipamentos característicos	Equipamentos característicos
.Reatores, Secadores, Cristalizadores, Colunas de destilação	.Bombas, Filtros, Trocadores de Calor, Centrífugas

Nas duas últimas décadas tem-se observado uma demanda crescente por produtos de "alta tecnologia" ou tecnologia de ponta que são normalmente fabricados em unidades não-contínuas, e com isso os engenheiros químicos começaram a se preocupar com aspectos novos deste tipo de unidade. Dentre estes novos aspectos pode-se distinguir:

- Natureza dinâmica das operações,
- As descontinuidades devidas ao início e término das etapas individuais de processamento implicam em:
  .Eventos dependentes do tempo, isto é, identificação de instantes de tempo em que ocorre uma descontinuidade conhecida;
  .Eventos dependentes do estado, isto é, identificação de instantes de tempo em que ocorre uma descontinuidade dependente do estado do sistema,
- A execução das etapas de processamento depende de "receitas" ,
- Cada etapa pode demandar muitos equipamentos,.
- A escolha e alocação de um equipamento a uma determinada tarefa é governada pela adequação, disponibilidade, prioridade e outras regras de uso,
- Existem restrições globais com respeito a recursos compartilhados tais como operadores e utilidades,
- Uso extensivo da armazenagem intermediária.

Os processos não-contínuos são, com respeito às áreas citadas e também com respeito à operação, normalmente mais complexos do que os processos contínuos, no entanto sua utilização vem aumentando. As razões para o uso de processos não-contínuos são basicamente duas:

- Razões Tecnológicas:
  . Viabilizam processos com tempos de residência longos;
  . Viabilizam processos de síntese complexos;
  . Exigem conhecimento mais limitado dos processos;
  . Quando as informações sobre "scale-up" são inadequadas.
- Flexibilidade:
  . Natureza multipropósito dos equipamentos permitindo a execução de múltiplas tarefas e múltiplos produtos;
  . São adequados para a produção de baixos volumes;
  . São adequados quando a demanda dos produtos é sazonal ou incerta;
  . Permitem absorção de incertezas do processo.

Por exemplo, em função das altas temperaturas empregadas na indústria siderúrgica e da impossibilidade de interligar seus equipamentos através de fluxo contínuo, a produção de aço é feita em unidades descontínuas (restrição tecnológica). Por outro lado, em função da sazonalidade da produção de leite e demanda de seus derivados, a indústria de processamento de leite opera, em geral, em regime semi-contínuo, garantindo assim uma menor ociosidade de seus equipamentos(flexibilidade).

2.2.1. Aspectos relevantes para o problema de scheduling

Nesta seção introduzem-se dois aspectos das plantas químicas batelada e semi-contínuas que tem grande importância na solução de problemas de seqüênciamento e scheduling de tarefas. Estes dois aspectos são a estrutura de processamento que caracteriza a operação da planta e o problema do compartilhamento dos recursos pelas diversas tarefas a serem

processadas ou executadas na planta.

2.2.1.1. Estrutura de Processamento

As estruturas de processamento podem ser classificadas em quatro categorias básicas, que são (Baker (1973),Graves(1975)):

1. 1 estágio/1 processador;

Neste caso cada uma das n tarefas a ser processada requer apenas um estágio sendo disponível apenas um processador.

2. 1 estágio/M processadores em paralelo

É idêntico ao caso anterior utilizando-se m processadores em paralelo. Estes processadores podem apresentar ou não produtividades idênticas.

3. Multiestágios em série(FLOW SHOP)

Deseja-se processar N tarefas em M estágios usando M processadores em série, e todas as tarefas são processadas nos M estágios na mesma ordem, isto é, o processador (i-1) sempre precede o processador i. Este tipo de estrutura recebe ainda a denominação,na Engenharia Química, de estrutura "multiproduto"; e na literatura da área de Pesquisa Operacional é também chamada de "estrutura de processamento baseada em estágios", já que a concepção básica de seu processamento é o fluxo unidirecional para todos as tarefas, sendo então especialmente indicada para aqueles casos em que haja forte semelhança entre as etapas de processamento entre as tarefas.

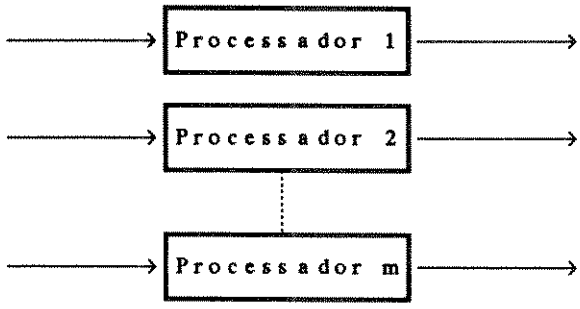
4. Multiestágios (JOB SHOP)

Neste caso, as rotas de operações de cada tarefa são diferentes, isto é, cada tarefa deve ser processada em k≤M estágios, mas a rota de processamento é diferente para cada tarefa. Este tipo de estrutura recebe ainda a denominação, na Engenharia Química, de estrutura "multipropósito"; e na literatura da área de Pesquisa Operacional é também chamada de "estrutura de processamento baseada em unidades", já que a concepção básica de seu processamento é a conectividade entre as diferentes unidades de processamento de maneira a flexibilizar diferentes rotas para o processamento de cada tarefa. Esta é também em grande parte a razão para o uso reduzido de estruturas job shop na Indústria Química, já que nesta é exigida uma ligação física complexa entre as unidades de processamento.

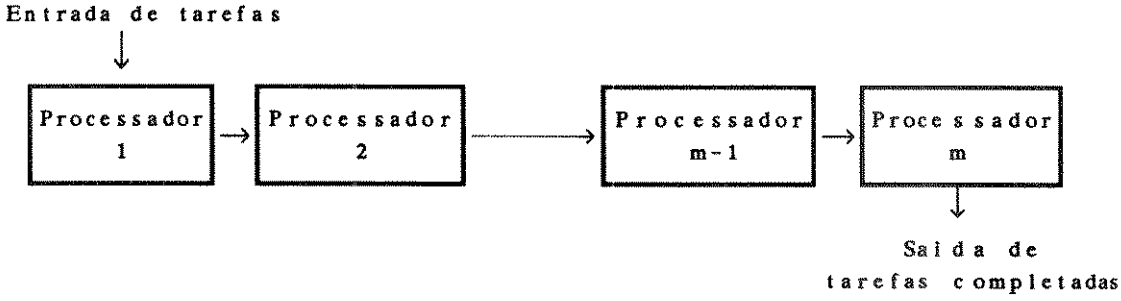
Estas quatro estruturas básicas de processamento podem ser visualizadas na Figura 2.1 (a-d).

Existem processos que obedecem uma estrutura básica do tipo flow shop isto é, processamento serial, mas que, no entanto, dispõem de mais de um processador para um ou mais estágios. Estes processadores múltiplos disponíveis nestes estágios podem ou não apresentar a mesma produtividade, isto é, exercem as mesmas funções mas com taxas de processamento (para o caso de processos semi-contínuos) ou dimensões (no caso de processos batelada), que podem ser iguais ou diferentes. Neste caso, a estrutura de processamento é chamada de flow shop generalizado(network flowshop) (Kuriyan(1987)). A possibilidade de se dispor de estágios com múltiplos processadores com produtividades iguais ou diferentes também pode ocorrer em uma estrutura job shop (Fox (1983), Mauderli e Rippin (1979)).

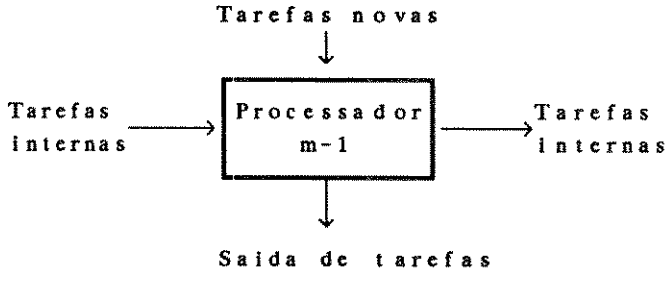
a)1 estágio/1 processador



b)1 estágio/m processadores



c)Representação do flow shop puro



d)Representação do job shop

Figura 2.1 - Representação das estruturas de processamento

De uma maneira geral os processos químicos não-contínuos, são realizados em estruturas multi-estágios. A escolha entre uma estrutura flow shop ou job shop depende fundamentalmente de dois fatores:

- Similaridade entre as rotas de processamento, isto é, quanto mais semelhantes forem as rotas de processamento, maior a probabilidade de se optar por uma estrutura flow shop;

- Número de produtos a serem produzidos na mesma unidade de processamento. As unidades não-contínuas são projetadas para produzir de 1 a 200 produtos. Quanto maior o números de produtos, mais complexas se tornam as atividades de projeto, simulação e planejamento da produção, já

que estes são problemas cuja natureza combinatorial é bem conhecida e, além disso para o caso de estruturas de processamento multi-estágios, são problemas NP-completo (Baker (1973), Conway (1972)). Obviamente à medida que cresce o número de produtos a serem produzidos na mesma unidade cresce também a possibilidade de serem agrupados em famílias de produtos similares (Dempster et al (1981), Bitran et al (1981)). A similaridade entre as etapas de processamento dos produtos favorece a escolha de uma estrutura flow shop. A estrutura job shop é geralmente reservada para aqueles casos especiais em que se deseja processar poucos produtos, que envolvam procedimentos de síntese bastante complexos e/ou incertos, e/ou baixa produção, e/ou demanda incerta.

A estrutura de processamento é normalmente definida na atividade de projeto, enquanto que para as demais atividades é um dado definido a priori que, embora não possa ser alterado,pode ser tratado de forma simplificada.

2.2.1.2. Recursos Compartilhados

A simultaneidade na execução de tarefas em estruturas seriais ou paralelas introduz imediatamente o conceito de compartilhamento de recursos. No caso de processos químicos flexíveis são considerados, num sentido amplo, como recursos compartilhados o tempo, equipamentos, matérias-primas, armazenagem, mão-de-obra e utilidades(vapor, água, eletricidade etc) (Reklaitis (1982), Rippin (1983)(1)).

No caso de problemas de Programação da Produção o tempo é o recurso compartilhado por excelência. O objetivo é,normalmente, partilhá-lo de maneira mais eficiente entre as várias tarefas de maneira que as restrições de disponibilidade e/ou utilização dos demais recursos necessários ao processamento sejam obedecidas.

Os equipamentos são recursos cujas regras de utilização e compartilhamento são ditadas pela estrutura, tipo de processamento e número de equipamentos disponíveis em cada estágio, de forma que em um problema de "scheduling" são tomadas decisões de alocação temporal de equipamentos ou conjunto de recursos a tarefas.

A importância das matérias-primas como recursos compartilhados depende fortemente da classe de problema de "scheduling", e como se verá mais a frente para a classe de problemas tratados neste trabalho admite-se que o fornecimento de matéria-primas não sofre qualquer tipo de restrição.

A armazenagem intermediária é um recurso de importância fundamental nos processos químicos e de natureza muito particular frente aos demais recursos compartilhados, cujas características intervêm de maneira decisiva na forma de processamento em estruturas multi-estágios. Ela é tratada normalmente em separado dos demais recursos compartilhados no problema de "scheduling".

Os recursos de mão-de-obra e de utilidades são normalmente disponíveis em quantidades limitadas no horizonte de tempo do "scheduling". São caracterizados por perfis de oferta e figuram no problema de otimização como "restrições". A importância relativa do perfil temporal de utilização destes recursos no problema de otimização determinará, em grande parte, a complexidade e forma de abordagem do problema de "scheduling". Estes recursos estão sujeitos, quando as condições permitirem, a flexibilizações

ou relaxamento das restrições de forma a acomodar soluções aproximadas. O relaxamento das restrições relativas a estes recursos pode não ser regido apenas por razões tecnológicas mas também econômicas, cabendo muitas vezes ao usuário a decisão de violar estas restrições.

2.2.1.3. A importância da armazenagem intermediária

A política de armazenagem intermediária em estruturas de processamento multiestágios, é normalmente definida na atividade de projeto, e obedece critérios de natureza tecnológica e econômica. As principais razões para se recorrer ao uso da armazenagem intermediária são (Leintein (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 "change-overs";
- No caso de processos semi-contínuos, permite que equipamentos adjacentes trabalhem a taxas de produção diferentes.

A definição de uma política de armazenagem intermediária não se restringe a definir o número e dimensão dos tanques, mas também onde colocá-los e porquê. Além disso, a forma como esse recurso será utilizado durante o processamento não depende exclusivamente de sua disponibilidade, mas também da definição de uma política de processamento dos produtos. Por exemplo, no caso de processamento de produtos instáveis, a existência ou não de armazenagem intermediária é irrelevante já que o processamento do produto deve prosseguir sem interrupção em qualquer um dos estágios, para que não haja comprometimento da qualidade do produto final ou até mesmo comprometimento da segurança de processo.

Estes dois aspectos, política de armazenagem e política de processamento, são frequentemente apresentados na literatura sob a denominação de "política de armazenagem", e é dentro deste contexto que são apresentadas a seguir as várias formas de armazenagem intermediária descritas na literatura para o caso de estruturas multiestágios. Os tipos de armazenagem são (Baker (1973), Karimi et al (1987)):

- Armazenagem dedicada : Neste caso, o tanque de armazenagem está associado a algum estágio de processamento, e não será compartilhado entre os produtos intermediários. De fato o estágio em questão é constituído pelo sistema tanque+processador.

- Armazenagem intermediária infinita (Unlimited Intermediate Storage - UIS). Significa que a armazenagem intermediária é suficiente para que, uma vez terminado o processamento dos produtos intermediários, eles sejam automaticamente transferidos para o estágio seguinte, se este estiver livre, caso contrário serão transferidos para a armazenagem intermediária. Pressupõe-se então que existe armazenagem suficiente para atender qualquer número de produtos em qualquer quantidade.

- Estrutura de processamento sem espera (No Wait- NW) em que os produtos intermediários, à medida que estão prontos, são imediatamente transferidos para os estágio seguinte e iniciado o processamento.

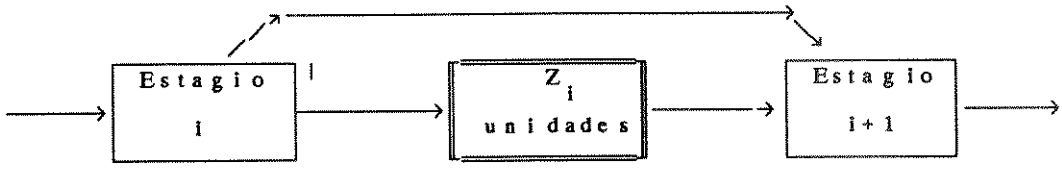
- Política de processamento sem armazenagem (No Intermediate Storage - NIS) em que o sistema não dispõe de armazenagem intermediária, de forma que o estágio pode atuar, se necessário, como armazenagem

intermediária. Note-se que, neste caso, o uso ou não da armazenagem intermediária dependerá realmente, não de sua existência física propriamente dita, mas sim da viabilidade técnica de um determinado equipamento poder atuar como tanque de armazenagem e do produto intermediário poder ser armazenado.

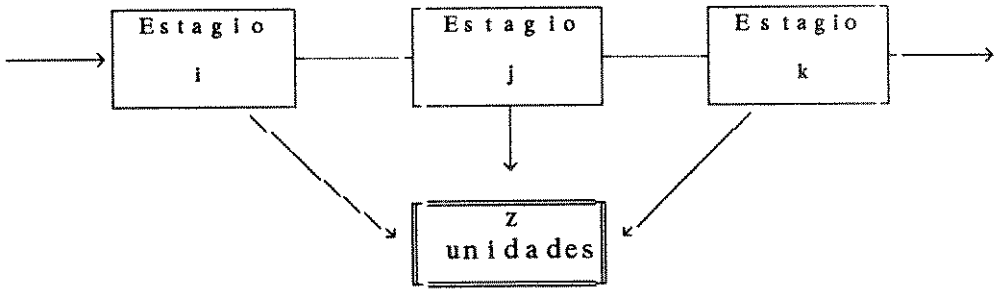
-Armazenagem intermediária finita (Finite Intermediate Storage - FIS) onde o produto, uma vez processado, somente será transferido para a armazenagem intermediária, se houver tanque livre,caso contrário permanece no próprio equipamento. Neste caso a armazenagem pode ser disponível interestágios, e portanto somente pode ser usada para armazenar o produto intermediário entre dois estágios sucessivos, ou compartilhada entre todos os estágios, quando pode ser usada após qualquer estágio de processamento.

Note-se que, salvo algumas exceções como o caso da indústria siderúrgica, tem-se políticas de armazenagem mistas, isto é, a armazenagem intermediária é finita e pode ser associada a restrições quanto à forma de processamento (NIS ou NW) dependentes dos produtos. Isto significa que a armazenagem intermediária poderá ser utilizada se estiver livre e se restrições tecnológicas permitirem.

Para ilustrar, na Figura 2.2 são apresentados os diversos tipos de armazenagem intermediária utilizados na indústria química.



a)Armazenagem interestágios



b)Armazenagem compartilhada entre estágios

Figura 2.2 - Tipos de armazenagem intermediária

2.3. Descrição geral do problema de projeto e operação de plantas químicas flexíveis

Em qualquer problema de processamento descontínuo estão sempre presentes três componentes:

- As condições que devem ser oferecidas para a produção dos diferentes produtos,
- As tarefas a serem executadas e as rotas para a produção dos diferentes produtos a partir de suas matérias-primas,

- Os equipamentos nos quais as tarefas de processo devem ser executadas.

Nos problemas de projeto e operação de plantas flexíveis assume-se que a seqüência de tarefas para cada produto é especificada a priori de forma que não se põe o problema de síntese de processo.

A dificuldade do problema de projeto e operação de plantas descontínuas reside no número de graus de liberdade para alocar as tarefas de processo aos equipamentos e decidir quando e de qual maneira os produtos devem ser produzidos. As tarefas a serem executadas são definidas e as condições que são oferecidas são, em geral, também conhecidas. No problema de projeto, o objetivo é a seleção das dimensões apropriadas do equipamentos de forma a satisfazer um conjunto de restrições e demandas, enquanto que o problema de Planejamento de Produção se preocupa em alocar e realocar os recursos de produção para atingir a demanda prevista. Entre as atividades de projeto e planejamento de produção,pode ser identificada uma gama variada de problemas. O número de graus de liberdade destes problemas são potencialmente maiores do que os dos processos contínuos (Grosmann et al (1979)), e, de modo a se permitir a comparação entre diferentes formulações de problemas e métodos de solução, é importante que estejam claras as hipóteses assumidas em cada caso.

2.3.1. Caracterização das operações descontínuas

As operações descontínuas são caracterizadas por (Rippin (1983)(1),(2)):

- Extensão em que a tarefa deve ser executada,
- Tempo necessário para executar a tarefa,
- Capacidade requerida.

Uma batelada de produto (tarefa) é produzida através da execução de uma seqüência de operações, sendo que o rendimento e a qualidade do produto são determinados pelo grau de conversão do conjunto de operações necessárias para atingir a transformação desejada. A freqüência de execução das diferentes bateladas é determinada pelos tempos necessários para a execução das tarefas; as quantidades que podem ser produzidas são determinadas pelas capacidades necessárias nas várias operações e as dimensões dos equipamentos disponíveis.

A extensão em que uma operação é executada é definida pelo estado das correntes que deixam o equipamento em relação às correntes de entrada ou alguma especificação externa.O desempenho é definido em termos de conversão de uma reação química, grau de separação, ou o alcance de uma temperatura etc. Se existem meios de especificar a priori estas medidas de desempenho, então o desempenho global da seqüência de operações pode ser calculado através de balanços de massa e energia, da mesma maneira que os realizados para processos contínuos.

O tempo necessário para a execução das operações é normalmente especificado antecipadamente. Se a integridade da batelada é preservada através de todos os equipamentos, então o intervalo de tempo no qual bateladas sucessivas podem ser produzidas será controlada pelo equipamento que demanda o maior tempo de execução de uma operação. Ele fixará o ciclo limitante do processo. Em configurações mais complexas onde o mesmo produto pode ser produzido simultaneamente tendo alguns equipamentos em comum para

serem usados consecutivamente por diferentes bateladas, o cálculo do período de ciclo é definido como o intervalo entre bateladas sucessivas.

A quantidade a ser produzida determinará a capacidade necessária para cada operação. A capacidade necessária por unidade de massa é chamada "size factor", e é normalmente expressa em termos de volume, mas outros tipos de unidades são também utilizadas. Para uma determinada estrutura de interligação de equipamentos, a quantidade de produto que pode ser processada será determinada pelo equipamento que possui o menor "size factor".(Grosmann (1985)).

A caracterização dos processos descontínuos, bem como a descrição dos problemas e restrições globais decorrentes da estrutura de processamento são apresentadas a seguir.

2.3.1.1. Definição do produto

O produto pode ser definido em quantidade ou tipo, além disso devem também ser especificadas as seguintes características:

- Estrutura do produto: alguns produtos dependem da disponibilidade de produtos intermediários para que possam ser produzidos. A estrutura em árvore ligará os intermediários e produtos finais, e fornecerá a seqüência prioritária de produção as quais, juntamente com a posição da estocagem, irão definir restrições temporais na seqüência de produção.

- Flexibilidade da demanda: em alguns casos os níveis de produção podem ser especificados em um intervalo, viabilizando a oportunidade de produzir produtos mais lucrativos.

- Prazo de entrega: o caso mais simples é aquele em que os produtos devem estar prontos no máximo até o final do período de planejamento. Outros perfis mais complexos também são possíveis até o caso limite de um perfil diário arbitrário de uma demanda contínua ou discreta em função do tempo.

- Incerteza no modelo de demanda: esta característica pode ser levada em conta através da proposta de um conjunto de cenários distintos, sendo que cada um deles apresenta um modelo de demanda diferente. Outra abordagem seria a incorporação destas incertezas no modelo através da adoção de modelos de demanda estocásticos.

2.3.1.2. Especificação das operações de processo

Uma tarefa de processo é um conjunto de operações, cada uma delas realizada em um tipo de equipamento. As tarefas de processo são caracterizadas por:

- Seqüência de operações: nos problemas de projeto o equipamento onde a tarefa deverá ser realizada pode ainda não estar especificado, mas tanto para a atividade de projeto quanto para a de planejamento, a seqüência de operações é fixada a priori. Uma seqüência apropriada de operações define um processo completo de produção.

- Modo de execução das operações de uma tarefa (contínuo, semi-contínuo ou batelada) .

- Desempenho da operação: em geral é fixado a priori, mas em função de modificações que podem ocorrer no processo, tais como decréscimo da atividade catalítica, incrustações em trocadores de calor, qualidade das matérias-primas etc, podem necessitar revisões periódicas através da repetição dos balanços de massa e energia com as devidas alterações de parâmetros do sistema.

- Tempo de operação das tarefas : depende da forma de execução da operação, e também está sujeito a alterações em função de alteracões do nível de desempenho da tarefa.

- Capacidade requerida: A relevância das mudanças na capacidade requerida para o desempenho de uma operação está relacionada com a dimensão do equipamento disponível para executá-la. Esta característica é especialmente crítica se o equipamento é um gargalo do processo.

- Incerteza na especificação das operações: são devidas especialmente aos seguintes fatores - variações na qualidade da matéria-prima, degradação antecipada do desempenho dos equipamentos ou efeitos do scale-up.

2.3.1.3. Especificação dos equipamentos

As especificações dos produtos e da seqüência de operações são, normalmente, rígidas. A complexidade dos processos descontínuos reside na variedade de configurações possíveis de equipamentos e a maneira como os produtos podem ser alocados, o que confere ao problema uma natureza combinatória . Em geral as configurações possíveis dependem da seqüência de operaçõs de cada produto ou tarefa e do tipo de operações a serem realizadas, enquanto que as possibilidades de alocação refletem restrições globais do sistema que serão apresentadas no item a seguir. Os equipamentos são especificados através de:

- Tipo: a especificação mais simples de tipo de equipamento é restringir cada tipo de operação a um único equipamento. No entanto, em função da estrutura de processamento pode existir mais de um equipamento ou processador habilitado a executar uma determinada operação. Neste caso pode ser necessário especificar se há alguma restrição ou preferência na alocação do equipamento. Pode também ocorrer o agrupamento de várias operações em um só equipamento.

-Número: No caso de planejamento da produção o número de equipamentos é fixo enquanto que na atividade de projeto um dos objetivos é determinar o número de equipamentos que deverão ser instalados.

- Dimensões : No caso do planejamento da produção é um parâmetro fixo, enquanto que na atividade de projeto o objetivo básico é dimensionar os equipamentos

- Possibilidade de interconexões: No caso geral todas as inter-conexões possíveis entre equipamentos poderiam ser estabelecidas, de forma que não há qualquer impedimento na alocação das operações aos equipamentos. No caso em que as interconexões são limitadas, são criadas restrições de alocação, que restringem o espaço de soluções do problema.

- Grupos de equipamentos operando em ou fora de fase.

2.3.1.4. Restrições dependentes da estrutura do sistema

Estas restrições podem ser divididas em:restrições do modelo de produção, restrições na alocação de tarefas aos equipamentos e restrições na implementação da seqüência de tarefas.

- Estrutura de produção (Rippin (1983)(1),(2), Mauderli et al (1979)): As estruturas de processamento são basicamente três: multiproduto, multipropósito e multiplanta. Em uma planta multiproduto, os produtos são produzidos sucessivamente numa seqüência de campanhas de produção. O fluxo de processamento é unidirecional e podendo existir uma ou mais rotas de produção para cada produto ou tarefa. Esta definição identifica uma estrutura multiproduto com um flow shop. Numa estrutura multi-planta duas ou mais plantas multi-produto operam em paralelo. Esta estrutura tem a vantagem de privilegiar que produtos com receitas e tempos de operação muito similares sejam agrupados e processados na mesma planta, provocando uma redução significativa da armazenagem intermediária, mas a custa de uma certa perda na economia de escala devido à multiplicação de equipamentos semelhantes. A estrutura multi-planta é identificada com o network flow shop. Numa planta multipropósito a produção de cada produto obedece diferentes rotas de produção tal como em uma estrutura job shop. As rotas de produção são em geral pré-definidas e a alocação do tempo de produção dos diferentes produtos pode ser feita por campanhas ou bateladas.

- Alocação de operações aos equipamentos: O procedimento mais simples é alocar cada operação a um equipamento. No caso de processos químicos é freqüente o agrupamento de operações em uma única operação se isto for tecnológicamente viável. Este procedimento ocorre com freqüência com operações individuais tais como mistura, aquecimento, reação e resfriamento.

- Não interrupção da seqüência de execução das operações de processo: A produção de alguns produtos pode requerer uma longa seqüência de operações. Em alguns casos pode ser apropriado considerar a interrupção da seqüência global de tarefas em sub-seqüências tecnologicamente factíveis, e que produziriam intermediários que atuariam como matérias-primas para a sub-seqüência seguintes.

2.3.1.5. Restrições operacionais e armazenagem

As restrições operacionais são devidas a:

- Limitações no compartilhamento de recursos.
- Instabilidades de produtos e/ou intermediários: Neste caso o produto deve ser processado de forma a prescindir do uso da armazenagem intermediária ou mesmo esperar no equipamento até que o equipamento seguinte na sua rota esteja livre.
- Limitações na disponibilidade de matéria-prima.

A armazenagem é dividida em dois tipos principais:

- Estocagem, responsável pelo armazenamento de matérias-primas e produtos finais, e acomoda mudanças no plano de produção de longo prazo;

- Armazenagem intermediária, responsável pela absorção de flutuações de produção (estoque ativo).

A incorporação destas restrições no nível de projeto ainda é bastante reduzida. No nível de planejamento e programação da produção, estas restrições são especialmente importantes para o problema do curto prazo, já que influenciam diretamente na factibilidade do "scheduling" (Grosmann et al (1983), Suhami et al (1982)).

2.3.2. O Problema de Projeto de Unidades Químicas Flexíveis

Qualquer unidade de processamento, quer seja contínua, não-contínua, flexível ou não, é dimensionada para uma determinada capacidade de produção; isto é, os equipamentos somente podem ser dimensionados a partir da definição de qual deverá ser a capacidade instalada.

No caso de unidades dedicadas, embora a atividade de projeto seja extensa e complexa, ela é, pelo menos em princípio, mais simples de ser executada do que a atividade similar para os processos flexíveis, visto que o projeto de unidades flexíveis envolve um número de graus de liberdade maior. Assim a atividade de projeto de unidades flexíveis engloba não só a definição dos fluxos de massa e energia envolvidos no processo, mas também a sincronização das operações em cada equipamento e sua interação com outras unidades na seqüência de produção.

Um dos primeiros problemas que se tem ao se iniciar a atividade de projeto de unidades descontínuas é com relação à definição da "capacidade da planta". É claro que a definição da capacidade da planta não é uma atividade de projeto, mas fundamental para sua execução,de forma que, com base em critérios exógenos à atividade de projeto, são definidos níveis de produção desejáveis que fornecerão subsídios para as demais atividades de projeto.

O programa de produção adotado para desenvolver a atividade de projeto, é normalmente muito mais simplificado do que os programas de produção reais, não se constituindo, portanto, base de comparação na análise de desempenho de planos de produção diferentes.

A atividade de projeto envolve três etapas básicas:

- Estudo de processo, quando são especificadas as "receitas" de processo, são definidas as possibilidades de interação entre linhas de produção, e o tipo de estrutura de processamento.

- Projeto preliminar, quando são definidas as quantidades de equipamentos necessários para a execução das tarefas e suas dimensões básicas.

- Projeto mecânico detalhado, quando são definidos e calculados detalhes mecânicos dos equipamentos.

Em função das da natureza de cada uma das atividades de projeto apresentadas acima, depreende-se imediatamente que o projeto preliminar é a atividade onde a definição de um plano de produção é importante.

Até recentemente a formulação do problema de projeto de plantas flexíveis era feita sem considerar os problemas de "scheduling" (Sparrow et al (1975)). A partir de 1975 foram desenvolvidos vários algoritmos que

incorporam problemas de "scheduling", mas cujas hipóteses acerca de política de armazenagem, compartilhamento de recursos etc, são bastante restritivas ou inexistentes (Grosmann et al (1979), Suhami et al. (1982), Grosmann et al. (1983), Vaselnak et al.(1987), Coulman (1989), Knopf et al. (1982)) . O problema de projeto sofre ainda de uma abordagem restrita em função da complexidade da formulação do problema "projeto+scheduling" e da inexistência de ferramentas eficientes para a sua resolução global.

Formalmente o problema de projeto básico de unidades flexíveis é colocado como otimizar as dimensões das unidades de processamento minimizando uma função objetivo, expressa normalmente em termos de custo. Embora formulações detalhadas da função objetivo possam incorporar custos anuais de investimento, amortizações, custos operacionais e custos associados a penalidades, devem ser considerados apenas os elementos chave que afetam as dimensões básicas dos equipamentos. Obviamente a incerteza na demanda e a seleção de um plano ótimo de produção, que são essenciais para avaliações de custo, não podem ser introduzidas no projeto básico.

A atividade de projeto de unidades flexíveis não se esgota no projeto de unidades novas; ao contrário, dada a sazonalidade dos perfis de demanda de produtos bem como ao desenvolvimento de novos produtos, as unidades não-contínuas e em especial as unidades em batelada, são frequentemente objeto de estudos para que sejam desenvolvidas atividades de projeto que permitam o desgargalamento (debotllenecking) das unidades, de forma a acomodar novos perfis de demanda, e a adição de novos produtos à linha de produção. Estas duas atividades, embora tenham objetivos diferentes, são, em essência, bastante semelhantes já que se baseiam na simulação para a identificação de estágios controlantes do fluxo de atividades (gargalos) e posteriormente recorrem à programação matemática para se proceder o dimensionamento de eventuais equipamentos que se deseje adicionar à estrutura de processamento. Esta atividade é classificada como Retrofit Design, cuja importância é apresentada no tópico 2.3.2.2

2.3.2.1. Exemplo de formulação de um projeto básico (Espuna et al (1989))

O problema em questão é calcular as dimensões básicas de uma planta química flexível, cuja estrutura de processamento é classificada como um "flow shop generalizado", isto é, será permitida a existência de equipamentos idênticos operando em paralelo em um mesmo estágio. Esta operação pode ser em fase ou fora de fase. Devem ser especificados:

- Número e quantidade dos produtos;
- Receitas da produção;
- Balanços de massa.

As hipóteses admitidas com relação à operação da planta são:

- Serão admitidas apenas campanhas de um só produto. A influência desta simplificação depende fortemente dos tempos de processamento e transferência entre equipamentos para cada produto, pois é ele que determina o intervalo de tempo em que dois equipamentos adjacentes estarão simultâneamente ocupados. A adoção de um modelo de produção considerando campanhas de um só produto pode também trazer conseqüências sobre o valor do makespan. Quando os tempos de processamentos são semelhantes para cada operação de cada tarefa, esta influência deve ser pequena. No entanto, quando os tempos de processamento das diversas operações das tarefas são

bastante diferentes, a influência pode ser significativa tal como ilustrado na Figura 2.3;

- As restrições em recursos compartilhados não são consideradas, o que equivale a admitir que a oferta é ilimitada;

- São consideradas apenas as "seqüências de permutação", isto é, aquelas seqüências em que a ordem de processamento das operações de cada tarefa é a mesma em todos os equipamentos. Também não é considerada a existência da armazenagem intermediária;

- Cada unidade de equipamento é utilizada apenas uma vez para o processamento do produto e, conseqüentemente será ao final liberada para o processamento da próxima tarefa da seqüência.

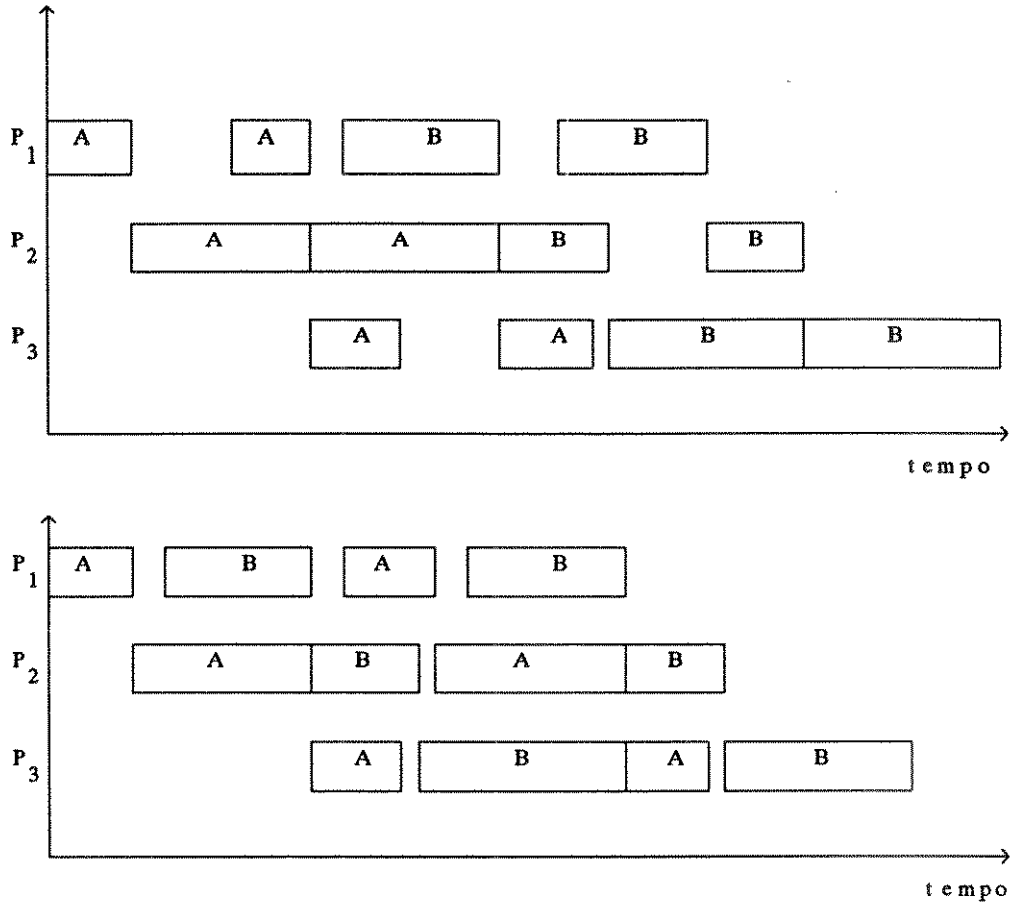


Figura 2.3 - Efeito do plano de produção sobre o makespan

Além das hipóteses acerca da operação da planta são também adotadas hipóteses adicionais referentes exclusivamente às limitações do projeto:

- O intervalo de dimensão dos equipamentos é pré-determinado;
- Os equipamentos em paralelo são idênticos;
- Não é feito o dimensionamento dos equipamentos semi-contínuos;
- A alocação das tarefas aos equipamentos é fixa.

A formulação matemática de um exemplo do problema é apresentada a

seguir e é composta de:

1)A função objetivo;

2)Relações entre custo e dimensão dos equipamentos;

3)Relações entre a quantidade processada e a dimensão mínima requerida. Dentre as variáveis normalmente empregadas para caracterizar a capacidade de plantas químicas descontínuas estão o "size factor" e o "batch size". O size factor ou fator de capacidade representa a dimensão característica do equipamento, em geral volume útil do equipamento, dividido pela massa total de produto que pode ser processada. O batch size representa a quantidade de produto, normalmente em massa, possível de ser processada em cada equipamento.

Função objetivo:

$$\min \sum_{j=1}^{m} [m_{j0} m_{jp}].[c_{j_1} + c_{j_2} V_j^{c_{j_3}}] + \sum_{k=1}^{L} m_{kp} [c_{k_1} + c_{k_2} . R_k^{c_{k_3}}] \qquad (2.1)$$

Restrições

$$V_j \geq \frac{S_{ij} B_i}{m_{jp}} \longrightarrow B_{i,max} = \min_i \left\{ m_{jp} \frac{V_j}{S_{ij}} \right\} \qquad (2.2)$$

$$F_{ij} \geq \frac{B_i \frac{D_{ik}}{R_k}}{m_{kp}} \longrightarrow F_{ij,max} = \min_{k \in K_{jF}} \left\{ \frac{B_i \frac{D_{ik}}{R_k}}{m_{kp}} \right\} \qquad (2.3)$$

$$E_{ij} \geq \frac{B_i \frac{D_{ik}}{R_k}}{m_{kp}} \longrightarrow F_{ij,max} = \min_{k \in K_{kE}} \left\{ \frac{B_i \frac{D_{ik}}{R_k}}{m_{kp}} \right\} \qquad (2.4)$$

$$P_{ij} = P_{ij_1} + P_{ij_2} . \left[\frac{B_{ij}}{m_{jp}} \right]^{P_{j_3}} \qquad (2.5)$$

$$t_{ij} = \frac{F_{ij} + P_{ij} + E_{ij}}{m_{j0}} \qquad (2.6)$$

$$T = \max_j \{F_{ij}, t_{ij}, E_{ij}\} \qquad (2.7)$$

$$H \geq \sum_{i=1}^{N} \frac{Q_i}{B_i} . T_i \longrightarrow SP = H - \sum_{i=1}^{N} \frac{Q_i}{B_i} T_i \qquad (2.8)$$

$$\underline{V}_j \leq V_j \leq \overline{V}_j \qquad (2.9)$$

$$\underline{R}_k \leq R_k \leq \overline{R}_k \qquad (2.10)$$

sendo:

m_{j0} : número de equipamentos operando fora de fase

m_{jp} : número de equipamentos operando em fase;

c_{k1}, c_{k2}, c_{k3} : coeficientes de custo;

V_j : volume do equipamento(dimensão);

R_k; taxa de processamento do equipamento semi-contínuo

E_{ij} :tempo para encher o equipamento do estágio j com o produto i

F_{ij} : tempo para esvaziar o equipamento do estágio j com o produto i

S_{ij} : "Size factor", definido como a dimensão característica do equipamento j por unidade de massa do produto i

B_i : "Batch size" do produto i

D_{ik} : duty factor para o produto i na unidade semi contínua k

T_i : período de ciclo limite para o produto i

P_{ij} : tempo de processamento do produto i na operação j

H : tempo total de produção disponível

Q_i : Produção requerida para o produto i

A expressão (2.2) indica que o batch size, isto é, a quantidade de massa de produto i que pode ser produzido, deve ser tal que o nível de produção desejado para cada produto é garantido. No nível ótimo de produção da planta, B_i irá corresponder, para cada produto, a $B_{i,max}$, de maneira a ter o equipamento limitante totalmente ocupado.

As expressões (2.3) e (2.4) fixam que os tempos para encher os equipamento (F_{ij}) e esvaziar os equipamentos (E_{ij}) serão limitados pela menor taxa de processamento envolvida, isto é, o maior tempo de transferência.

Através da expressão (2.5) são calculados os tempos de operação nos equipamentos batelada. Nos níveis ótimos de produção da planta, os tempos de transferência e processamento são identificados com seus valores mínimos factíveis, $F_{ij,min}$, P_{ij} e $E_{ij,min}$. O tempo total de operação será a soma destas três parcelas (equação 2.6).

A expressão (2.7) estabelece que o período de ciclo para a batelada do produto i não pode ser menor que o necessário para cada uma das operações envolvidas no processamento.

Uma vez determinado o período de ciclo e o "batch size" a expressão (2.8) estabelece que o tempo total de produção não pode exceder o tempo total disponível (H).

Este é um problema chamado de NPL (Non Linear Problem), e pode ser resolvido através de diferentes técnicas tais como GRG (Generalized Reduced Gradient)(Bazaraa (1979)). No entanto deve-se ter em conta que, mesmo para problemas de pequeno porte estes algoritmos consomem muito tempo de computação.

Uma generalização já proposta por diferentes autores, é a incorporação de restrições que limitam as dimensões dos equipamentos a valores padrão. Neste caso, há a transformação de um problema NPL em um problema MINLP (Mixed Integer Non Linear Problem), que exige o recurso a técnicas mais complexas de resolução devido à introdução de restrições de "integralidade" já que as dimensões factíveis dos equipamentos são variáveis discretas. Uma das técnicas viáveis para a resolução do problema é a decomposição de Bender (Geoffrion (1972)).

O problema de projeto de unidades flexíveis foi apresentado numa de suas versões mais simples. A generalização deste problema não será feita através da incorporação de hipóteses mais complexas sobre a operação da planta, mas sim através do acoplamento de um algoritmo de planejamento de produção para viabilizar um procedimento de otimização.

2.3.2.2. O problema do Retrofit Design : Exemplo (Vaselenak et al.(1987))

Esta atividade tem lugar sempre que ocorram variações significativas no perfil de demanda ou se deseja adicionar um novo produto à planta.

Em geral um critério para decidir se um ou mais novos equipamentos serão adicionados à planta é comparar o custo do(s) equipamento(s) e o retorno de investimento devido ao aumento da produção. É claro que se o custo é maior do que o retorno, a solução é rejeitada.

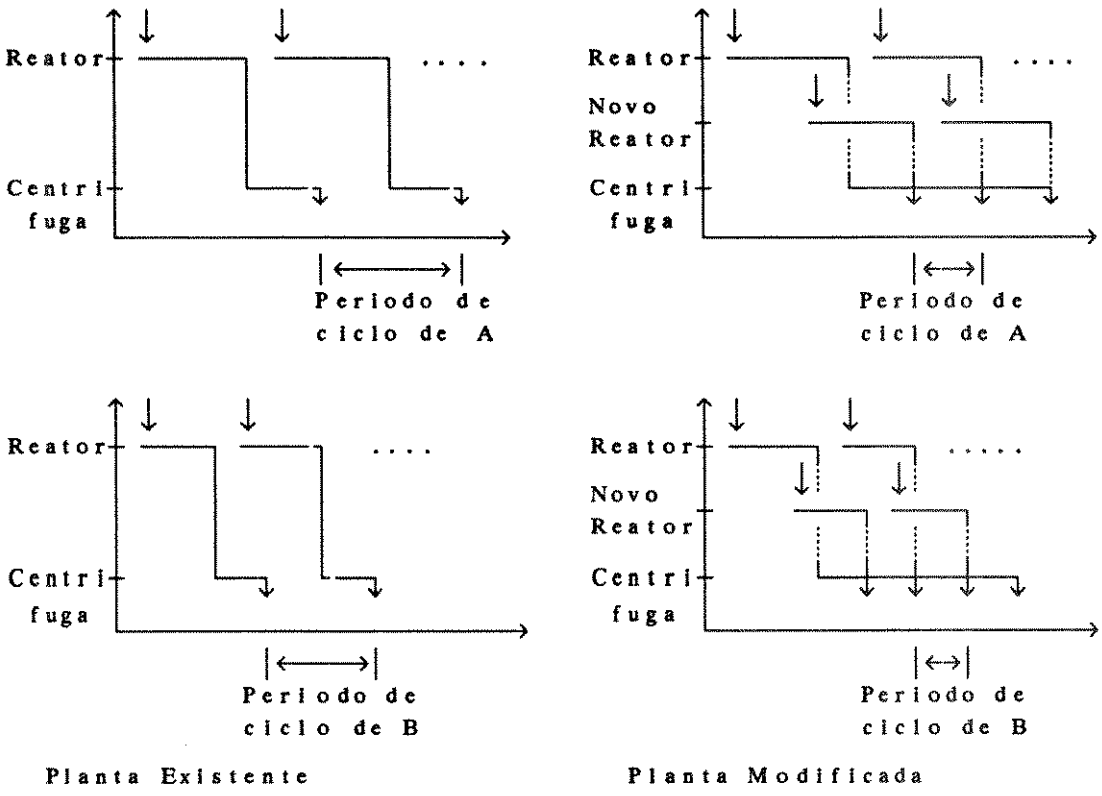
Para se visualizar os ganhos que podem decorrer destes estudos, são apresentados na Figura 2.4 duas situações clássicas em que a adição de equipamentos em paralelo implicam em ganhos substanciais. No caso a) a adição de um reator em paralelo operando fora de fase (out of phase)provoca um decréscimo do período de ciclo e consequentemente permite um aumento da freqüência de bateladas e portanto de produção, ilustrando um caso de desgargalamento. O caso b) ilustra uma situação em que a adição de um equipamento em fase aumenta o "batch size" e portanto permite uma melhor utilização da capacidade da planta.

Do ponto de vista da formulação matemática e hipóteses simplificativas, o problema do retrofit é bastante semelhante à do projeto básico.

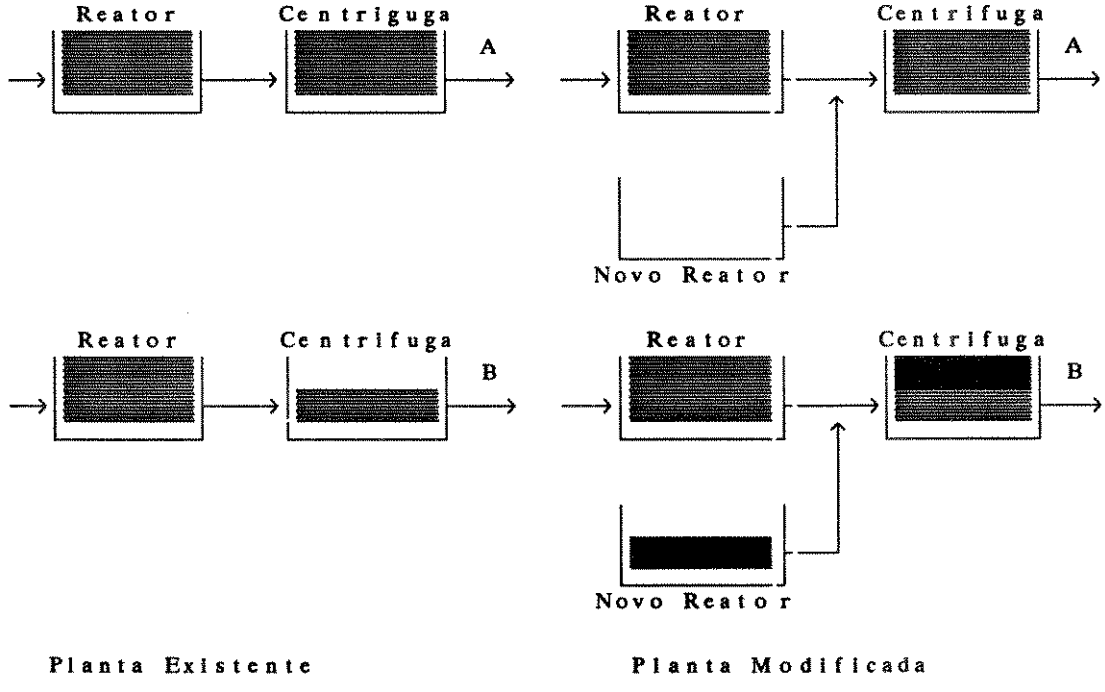
2.3.3. Simulação de Unidades Químicas Flexíveis

A atividade de Simulação de Processos Flexíveis é bastante complexa já que, em geral, as plantas químicas flexíveis são operadas com base em "regras", estão sujeitas à flutuações aleatórias e mudanças discretas, e as interações entre os diferentes componentes da planta são complexas (Felder (1979), Mauderli et al (1980), Sparrow et al. (1974), Joglekar et al (1984), Joglekar et al. (1987)).

A simulação tem como objetivo avaliar diferentes alternativas antes que recursos sejam comprometidos, para a redução dos riscos na tomada de decisões, aumentar a produtividade, e aumentar o lucro. Por estas razões a simulação é uma ferramenta poderosa e indispensável para auxiliar as atividades de projeto planejamento e controle.



a - Exemplo de Desgargalamento



b - Expansão do Batch Size

Figura 2.4 - Exemplos de aplicação da atividade de Retrofit

O cenário em que é desenvolvida a atividade de simulação de processos flexíveis pode ser definido como:

Dado: uma planta multipropósito/multiproduto
-Com operações batelada e/ou semi-contínuas;
-Com uso intensivo de trabalho, difícil de controlar;
-Sujeita a incertezas e condições difíceis de serem reproduzidas;
-Sujeita a atrasos planejados e não planejados;
-Com unidades sujeitas a ociosidades e bloqueios;
-Com gargalos difíceis de serem determinados;
-Com interações complexas entre processos, recursos compartilhados e armazenagem.

Deseja-se responder uma ou várias das seguintes questões:

-Quais são os gargalos da planta;
-Em qual extensão a produtividade é limitada pela capacidade do equipamento, pela armazenagem intermediária,pela não disponibilidade em níveis mais elevados dos demais recursos compartilhados e pela operação ineficiente da planta;
-Qual efeito pode ocorrer pela adição de um equipamento, operador, ou mudança nos procedimentos de operação da processo;
-Qual o efeito destas mudanças na produtividade da planta;
-Quais mudanças devem então ser feitas.

O caminho então é simular e otimizar cada processo em todas as áreas em que ele seja desenvolvido considerando:

-Variações, incertezas, e atrasos;
-Ociosidades, bloqueios e disponibilidade de operadores;em um período de tempo longo o suficiente para gerar estatísticas representativas do desempenho do processo. Estes dados devem então ser comparados com estatísticas da planta. Se estas comparações não forem satisfatórias deve-se coletar mais dados para melhorar a simulação.

2.3.3.1.Exemplo do tipo de resultados obtidos da simulação em um processo flexível (Felder - comunicacao privada)

Na Figura 2.5 é apresentado um processo típico para o qual se deseja estudar eventuais gargalos na operação.

São conhecidos:

-Durações de cada etapa do processo;
-Seqüência das etapas de processo, incluindo manutenção;
-Número de unidades de cada tipo;
-Fluxos de massa em todas as correntes contínuas;
-Número de operadores em cada área de produção, e os números necessários para cada etapa;
-Condições de alarme;
-Procedimentos a serem seguidos sob condições normais de operação, quando ocorrem bloqueios, e quando ocorrem condições de alarme;
-Estado inicial do sistema incluindo níveis dos tanques;
-Intervalo de tempo para a simulação.

As respostas do sistema desejadas são:

-Taxas de produção;
-Utilização dos equipamentos e dos recursos compartilhados;
-Freqüência e duração dos bloqueios e atrasos;
-Temperaturas, pressões, níveis, fluxos etc
-Tabelas, gráficos e histogramas, relatório final.

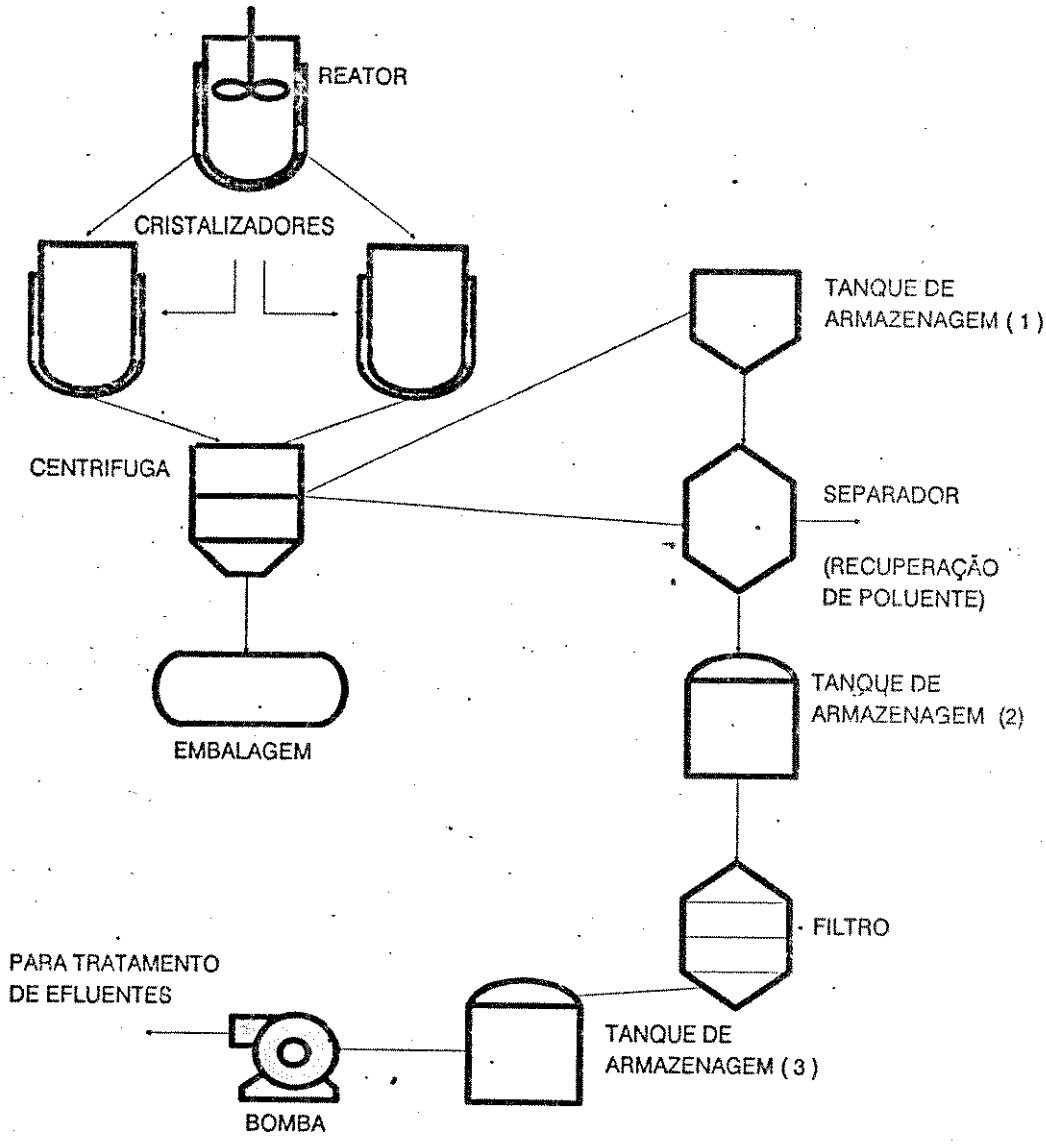


Figura 2.5 - Exemplo de unidade para a simulação

Na Tabela 2.5 é apresentada a comparação entre os dados de operação da planta e o resultado da simulação através do modelo proposto. A concordância entre estes resultados mostra que o modelo adotado é adequado para fazer projeções futuras da planta.

Na Tabela 2.6 são apresentados os resultados da simulação para outros cenários em que sejam adicionados alternadamente e simultaneamente equipamentos idênticos aos candidatos potenciais a gargalos do processo, e o ganho resultante destas alterações em termos de números de bateladas produzidas no mesmo intervalo de tempo.

Na Tabela 2.7 é apresentado o efeito da adição de tanques de armazenagem em termos de número de eventos atrasados. Os eventos atrasados são calculados com base na distribuição temporal dos eventos que é obtida para a configuração real da planta e a distribuição temporal que seria obtida caso a disponibilidade de armazenagem intermediária fôsse infinita.

Tabela 2.5 - Comparação entre os dados da planta e a simulação

Bateladas por semana Planta=16		Simulação=16
Nível de utilização de equipamentos (% de tempo em uso)		
	Planta	Simulação
* Reator	73.4	76.6
* Centrífuga	75.2	77.9
* Cristalizadores	76.2	71.4
Separador	62.5	61.3
Filtro	24.4	19.4
Embalagem	14.8	15.4

* Candidatos a gargalos de processo

Tabela 2.6 - Efeito das modificações propostas para o desgargalamento

Efeito das modificações de processo na produção semanal**			
Reatores	Cristalizadores	Centrífugas	Bateladas produzidas*
1	2	1	16
1	3	1	17
2	2	1	16
2	3	1	18
1	2	2	16
1	3	2	17
2	2	2	16
2	3	2	19

*No de operadores = 3

** Foi admitida armazenagem infinita

Tabela 2.7 - Efeito do tanque de armazenagem (1) na produção semanal*

Capacidade armazenagem	no. de bateladas	no. de eventos atrasados	atraso total (tempo)
0	11	11	124
140	19	17	42
280	21	16	22
420	22	11	12
560	22	9	6
700	22	7	6
840	22	8	7
ilimitada	22	0	0

* Sistema: 2 reatores, 3 cristalizadores, 2 centrífugas, 5 operadores

Este exemplo demonstra a importância da simulação como ferramenta para as atividades de projeto de plantas flexíveis pois permite identificar os gargalos de processamento de forma a orientar a atividade de retrofit ou eventualmente substituí-la.

2.3.4. Gerenciamento da Produção

O problema de Gerenciamento da Produção onde está incluído o Planejamento e Programação da Produção, é definido da seguinte forma: "Alocar os recursos disponíveis (tempo, trabalho, matérias-primas etc), para satisfazer a demanda do mercado em um período extenso de tempo (horizonte multi-período) da maneira mais eficiente". Significa então que conhecida a demanda dos produtos se deseja:

- Dimensionar a produção;
- Definir os perfis temporais dos recursos necessários;
- Definir os níveis de estoque;
- Definir as rotas e as alocações dos equipamentos;
- Definir a seqüência de produção dos produtos;

de forma que seja minimizado um critério de custo ou desempenho do sistema.

O problema global de gerenciamento da produção tal como apresentado é extremamente complexo e não foram até o momento, apresentadas abordagens adequadas para sua solução.

Existem duas abordagens para o tratamento do problema: abordagem em nível único e abordagem hierárquica (Reklaitis (1983), Bensana (1986), Korf (1987), Bitran(1981), Sacerdoti (1974),Dempster(1981)).

Na abordagem em nível único, tanto as atividades de planejamento quanto as atividades de programação e seqüênciamento de produtos são consideradas simultâneamente. O objetivo é determinar as quantidades de produtos a serem produzidas, os recursos necessários, alocação de produtos

aos equipamentos e a seqüência de produção em cada período de tempo a fim de minimizar os custos de produção, armazenagem e utilização dos recursos. Se os diversos custos são independentes da seqüência de produção e a alocação das operações aos equipamentos é flexível, o problema pode ser formulado como um problema de Programação Linear (PL). Contudo a incorporação de detalhes de seqüênciamento exige a introdução de variáveis binárias (0/1) ou inteiras, transformando um problema de PL em, no mínimo, um problema de programação inteira mista, o qual é combinatorialmente complexo e pode ser resolvido para problemas de pequena dimensão.

Na abordagem hierárquica, o problema de Gerenciamneto da Produção é decomposto em pelo menos dois subproblemas(Baker (1972), Reklaitis (1979)):

- Problema de Planejamento (Longo Prazo): que se ocupa em definir qual o nível de atividade necessária no período, o nível de recursos necessários e os níveis de estoque;

- Problema da Programacao (Curto Prazo): que se ocupa em definir a seqüência de produtos, a alocação de equipamentos e a distribuição temporal das operações.

Um dos maiores problemas na abordagem hierárquica é definir de forma clara os contornos de competência de cada um dos subproblemas. É evidente que problemas de curto prazo somente se põem quando os problemas de longo prazo estão equacionados e são conhecidas as interrelações entre os dois níveis. A definição do tipo de problema do curto prazo é fundamental para estabelecer quais variáveis devem ser calculadas ou especificadas nos níveis anteriores.(Dempster et al. (1981), Bitran et al (1981))

O Planejamento e a Programação da Produção são duas atividades fortemente relacionadas(Birewar e Grosmann (1983)). Devido à complexidade em se desenvolver esta atividade em um nível único, tem-se privilegiado a abordagem hierárquica do problema, através da divisão nos dois sub-problemas apresentados acima.

As decisões que normalmente competem ao nível do Planejamento representam as decisões táticas, dentre elas a expansão de utilidades ,expansão da capacidade instalada, mudanças de rotas de produção, definição dos níveis de atividades etc. No nível de Programação da Produção são privilegiadas as decisões operacionais, cujos limites de factibilidade são definidos no nível de Planejamento.

2.3.4.1. O Problema de Planejamento de Longo Prazo

O Planejamento tem por objetivo responder às seguintes questões:

- Qual produto ou serviço deve ser produzido ou executado;
- Qual o nível de atividade;
- Quais recursos devem ser colocados à disposição para a execução das tarefas;

sendo conhecidos:

- O número de produtos, suas receitas e recursos de produção;
- Equipamentos disponíveis e suas dimensões;

- Tempos de operação;

de forma a maximizar o lucro ou minimizar o "makespan" (tempo total para a execução do plano de produção). Neste nível não são tomadas decisões de seqüênciamento de produtos e não são necessárias informações temporais detalhadas do plano de produção (Carta de Gantt) (Shimizu et al. (1985), Sahinidis (1989)).

Exemplo de abordagem de um problema de Planejamento (Mauderli e Rippin (1979))

O exemplo em questão foi desenvolvido por Mauderli e Rippin, e o objetivo é determinar o plano de produção ótimo de forma a maximizar o lucro ou minimizar o makespan. A planta é formada por 5 reatores (R_1 a R_5) nos quais se deseja produzir 3 produtos diferentes (A,B,C). São permitidas quaisquer interconexões entre os equipamentos. Na Tabela 2.8 são apresentadas as dimensões de cada um dos equipamentos, e na Tabela 2.9 são apresentadas as operações que devem ser executadas para que cada produto (ou tarefa)seja completado, quais equipamentos podem ser utilizados, os tempos de operação para cada tarefa e o size factor ou fator de capacidade (S_{ij}). O usuário deve especificar as quantidades mínimas e máximas aceitáveis para a produção de cada um dos produtos.

A estratégia de solução do problema é desenvolvida em duas etapas. Na primeira etapa é utilizado um procedimento cujo objetivo é a determinação de campanhas de produção dominantes. As campanhas de produção são aquelas cujas combinações dos equipamentos conduzem à máxima produtividade. A segunda etapa consiste em determinar quais destas campanhas dominantes devem ser executadas e em que número para atingir a produção dos produtos desejados nos limites pré-fixados de forma a maximizar o lucro ou minimizar os custos. Nesta etapa a solução é encontrada através da solução de um problema de programação linear cujas equações não são apresentadas pelos autores.

A seguir são apresentados rapidamente os principais passos propostos pelos autores para definir as campanhas de produção dominantes.

Tabela 2.8 - Dimensões dos equipamentos

Reatores	R_1	R_2	R_3	R_4	R_5
Dimensão(l)	1000	2000	3000	4000	5000

Tabela 2.9 - Dados dos produtos para a construção de rotas de produção

Produtos	Operações	Equipamentos	Tempo de Processamento(h)	Size Factor (l/kg)
A	T_1	R_1 a R_5	2	1,0
	T_2	R_1 a R_5	5	2,0
B	T_1	R_1 a R_5	4	1,5
	T_2	R_1 a R_5	3	3,0
C	T_1	R_1 a R_5	3	2,0

1ª passo) Construção das rotas de produção: consiste em combinar os equipamentos disponíveis para a execução de cada tarefa de modo a ser possível a produção do produto desejado. A capacidade de produção numa rota é definida pelo equipamento de menor capacidade de produção. Por exemplo, para o produto A pode-se construir 20 rotas, ou seja, combinações dos cinco reatores 2 a 2. Na Tabela 2.10 são apresentadas algumas rotas possíveis para a produção de A.

Tabela 2.10 - Algumas rotas possíveis para a produção de A

nº da rota	T_1	T_2	Capacidade do estágio limitante { min V_j/S_{iA} }
1	R_5	R_4^+	2000
2	R_4	R_5^+	2500
3	R_3	R_5^+	2500
4	R_5	R_3^+	1500

O sinal "+" indica o estágio limitante.

2ª passo) Expansão das rotas: consiste em estudar as possíveis ampliações da capacidade de produção pela adição de equipamentos em paralelo.

3ª passo) Construção de linhas de produção: a linha de produção mais simples é constituída pela operação cíclica das rotas de produção. Uma alternativa consiste em compatilhar equipamentos através da operação fora de fase dos equipamentos, o que provoca a redução do período ocioso. Um critério para a comparação entre diferentes linhas de produção pode ser a produção média.

4ª passo) Estabelecimento das campanhas de produção dominantes: uma

campanha de produção consiste na operação simultânea de diferentes linhas de produção não sobrepostas, produzindo um ou mais produtos. As campanhas dominantes são aquelas geradas a partir de combinações de linhas de produção que, usando os mesmos equipamentos, apresentam maior produtividade.

Após a definição das campanhas dominantes, isto é, das campanhas que apresentam maior produtividade, o problema de otimizar o plano de produção é formulado como um problema de PL.

2.3.4.2. O Problema de Planejamento de Curto Prazo (Scheduling)

No nível de Planejamento de Longo Prazo é determinado o cenário de produção (níveis de produção e disponibilidade de recursos) e, como conseqüência, são estabelecidos os parâmetros para o problema de Planejamento de Curto Prazo (Planejamento e Programação da Produção)(Baker(1973)).

O problema de Programação da Produção (SCHEDULING) em flow shops é definido da seguinte maneira:

Sendo dados:

a_1 - N produtos e suas receitas;

a_2 - Os prazos de entrega dos produtos;

a_3 - A natureza da armazenagem intermediária e a política de processamento;

a_4 - A estrutura de processamento e os perfis temporais de oferta dos recursos necessários à execução das tarefas

a_5 - O critério de custo ou desempenho do sistema;

Determinar:

b_1 - A seqüência de produção;

b_2 - A alocação dos equipamentos aos produtos;

b_3 - Os instantes de início e fim das operações (Carta de Gantt).

A definição do número de produtos define uma dimensão básica do problema. A dimensão do espaço de busca está fortemente relacionada com as chamadas restrições de precedência, que definem um ordenamento parcial na execução das tarefas. As restrições tecnológicas de precedência são bastante comuns na indústria química já que frequentemente alguns produtos são produtos finais e ao mesmo tempo intermediários para a fabricação de outros produtos. Além da informação acerca da interdependência dos produtos, outros informações importantes devem constar das receitas de cada produto tais como:

- Tempo de processamento que especifica o tempo que o processador estará inteiramente dedicado a uma tarefa;

- Tempo de transferência que especifica o tempo necessário para transferir o produto entre os equipamentos que executam operações sucessivas e portanto define o período em que os equipamentos estão

simultâneamente ocupados. É importante notar que, na existência de armazenagem intermediária e caso ela seja solicitada, o produto deve ser transferido para e da armazenagem. Este tempo de transferência pode ser igual ou diferente do tempo de transferência entre equipamentos e portanto deve constar da especificação do produto;

- Tempo de preparação dos equipamentos que especifica o tempo que o processador está bloqueado , e no seu final o processador está preparado para receber outro produto. Este tempo pode ser dependente da seqüência;

- Natureza dos produtos intermediários. No caso de processos químicos é bastante comum a existência de intermediários instáveis de tal forma que, terminado o processamento deste produto, ele precisa ser imediatamente transferido para o processador seguinte. Deve-se então especificar ao final de cada estágio de processamento se o produto intermediário é ou não estável. Normalmente, como resultado desta análise são definidos os processadores que devem ser, para cada produto, agrupados de maneira a formar subconjuntos de processadores que devem operar sem espera entre os estágios de forma a garantir a integridade do produto final. Na Figura 2.6 é apresentado um exemplo de construção de blocos estáveis para o processamento. Em função das características físico-químicas dos produtos químicos é bastante comum a subdivisão de uma linha de processamento em "blocos" do tipo FIS, NIS e ZW (Karimi 1989).No caso limite em que todos os intermediários de um produto forem instáveis, este produto deve ser processado no regime NW (No Wait).

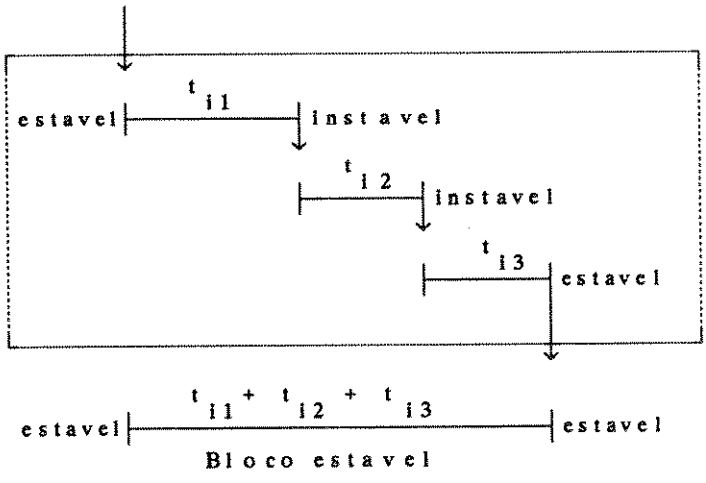


Figura 2.6 - Exemplo de contrução de blocos de processamento estáveis

- Quais recursos são necessários para a execução das tarefas e quais os perfis temporais de demanda de cada um dos recursos associados a cada tarefa.

Para o caso da Indústria Química predomina a estrutura de processamento tipo flow shop, com as seguintes características:

- O processamento das operações de cada tarefa é feito sem interrupção, e a ordem de execução das operações de cada tarefa é a mesma em todos os processadores. Isto significa que conhecida a seqüência de tarefas, estará igualmente definida a seqüência das operações em cada

processador. Estas seqüências são chamadas seqüências de permutação e reduzem a ordem do problema de $N!^M$ a N!.

- Uso intensivo da armazenagem intermediária. A armazenagem é normalmente limitada. A armazenagem intermediária está intimamente relacionada com a natureza dos produtos a serem processados. Não havendo restrições de estabilidade de intermediários, a política de utilização da armazenagem será definida pela distribuição e número de tanques de armazenagem intermediária. Deve-se no entanto ter em conta que aarmazenagem intermediária, embora seja um recurso a ser compartilhado pelas diferentes tarefas a serem processadas, apresenta algumas características diferentes dos demais recursos compartilhados. Trata-se de um recurso disponível cujo perfil de demanda não é definido a priori antes do processamento, isto é, a utilização da armazenagem intermediária não está associada a um modelo de demanda. A sua utilização está associada a aspectos de custo, facilitação de change-overs etc

-Uso intensivo de recursos compartilhados, especialmente utilidades (vapor, água de refrigeração etc).

-As conexões entre equipamentos são limitadas. As transferências entre equipamentos adjacentes não são instantâneas de forma que dois equipamentos adjacentes estarão simultaneamente durante a transferência. Para fins de scheduling estes tempos são normalmente associados aos tempos de processamento.

A definição da estrutura de processamento como sendo serial em estágios é feita, como se viu, antes da etapa de projeto. No entanto, se a estrutura for do tipo "network flowshop" serão necessárias algumas informações adicionais pois neste caso cada estágio poderá ser formados por um ou mais processadores idênticos ou não. As informações necessárias são informações do tipo:

- Quantos processadores estão disponíveis em cada estágio;
- Se estes processadores são, com respeito às características de processamento(volume, taxa ou tempo de processamento) idênticos ou não;
- Quais interconexões existem ou são possíveis entre os processadores de estágios consecutivos;
- Se todas as tarefas podem ser igualmente executadas em qualquer processador de cada um dos estágios ou se há algum tipo de restrição ou preferência de uma tarefa com relação a um processador.

Nos problemas de scheduling os elementos centrais são tarefas e recursos e processadores são também compartilhados. No entanto no caso do flow shop em que são exploradas exclusivamente as seqüências de permutação, além da fixação natural da direção de fluxo pela estrutura de processamento, a ordem de execução das operações em cada processador também está espaecificada, impondo naturalmente que, uma vez definida a tarefa a ser iniciada, a prioridade no uso do processador está definida e não pode ser alterada. Neste caso não se põe o problema de demanda simultânea de diferentes tarefas pelo recurso processador. Existe sim uma demanda simultânea, e portanto a possibilidade de geração de conflitos temporais, pelos demais recursos compartilhados tais como, no caso dos processos químicos, vapor, mão-de-obra, armazenagem intermediária etc. Para a construção de programações factíveis é necessário que sejam conhecidos, além dos perfis de demanda que caracterizam as tarefas, os perfis de oferta que caracterizam os recursos.

A teoria do scheduling está relacionada com modelos matemáticos, e portanto deve-se procurar descrever a estrutura dos problemas numa forma matemática concisa. A função objetivo deve ser composta em princípio, de todos os custos do sistema que dependam das decisões de seqüenciamento de tarefas e alocação de recursos. Todavia alguns destes custos são difíceis de serem avaliados na prática, ou mesmo de serem identificados completamente. De fato a maior parte dos custos de curto prazo são difíceis de serem isolados e aparecem fixos. Freqüentemente medidas de desempenho do sistema relacionadas com o custo(atraso na execução de tarefas, tempo ocioso do processador etc) podem ser usadas como substitutas do custo total do sistema. Uma das medidas de desempenho mais freqüentemente utilizadas é o makespan, que reflete o tempo total de execução das tarefas.

A organização adequada destas informações disponíveis e desejadas conduz à formulação de um problema de otimização cujo desenvolvimento é feito através das seguintes etapas:

1ª etapa) Definição da função objetivo: No mínimo uma função objetivo deve ser otimizada e é ela que define o objetivo a ser atingido, como por exemplo minimização do makespan, minimização do número de tarefas atrasadas etc.

2ª etapa) Definição das restrições: As restrições compreendem o modelo do processo e mostram como as variáveis envolvidas no processo afetam o critério de desempenho, e ainda definem a região factível para a busca da solução no espaço composto pelas variáveis de decisão. Os recursos são caracterizados em termos de sua capacidade e os tipos de operações que podem executar e as tarefas são caracterizadas pela quantidade de recurso necessário à sua execução, sua duração, quando ela deve ser iniciada e quando pode estar terminada. Além disso um conjunto de tarefas pode também ser caracterizado em termos de restrições tecnológicas (restrições de precedência).

3ª etapa) Busca da solução: Selecionar a melhor solução por métodos eficientes. A teoria do scheduling envolve um número variado de técnicas para a resolução dos problemas tais como procedimentos combinatórios, técnicas de simulação, progrmação matemática, soluções heurísticas etc. A seleção da técnica apropriada depende, dentre outros fatores, da complexidade do problema, da natureza do modelo, da escolha do critério de desempenho e em alguns casos pode ocorrer a combinação de diferentes técnicas de resolução.

A solução de um problema de scheduling é,em resumo, a busca de respostas a duas questões básicas:

1. Quais equipamentos devem ser alocados para executar cada tarefa;
2. Quando a tarefa deve ser executada;

ou seja envolve simultâneamente decisões de alocação e decisões de seqüenciamento.

As características básicas de cada problema conduzem a diferentes problemas de scheduling mais ou menos complexos. Um panorama geral dos diferentes tipos de problemas e diferentes propostas de solução serão apresentadas no Capítulo 3.

2.4. Conclusões

Neste Capítulo procurou-se dar uma visão geral dos problemas que afetam o projeto, planejamento e programação da produção das unidades químicas flexíveis, com especial atenção ao caso do processamento multiestágios serial (flow shop), como forma de introdução mais detalhada ao problema da programação de curto prazo, normalmente chamada de scheduling.

Como resultado principal desta apresentação é importante ressaltar que, em se tratando de unidades flexíveis (químicas ou não), muitos destes problemas, em especial os de projeto, são resolvidos quer a custa da adoção de um conjunto de hipóteses simplificadoras quer a custa da adoção de modelos de demanda simplificados. A conseqüência imediata é que não existe uma capacidade de produção da planta pré-definida, sendo necessário analisar sempre a possibilidade de atendimento do perfil de demanda requerido e para tanto deve-se dispor de um sistema de análise de capacidade e busca de soluções para o scheduling capaz de absorver constantes modificações na demanda e oferta de recursos, bem como na demanda de produtos. Como consequência o problema de scheduling em unidades flexíveis é geralmente resolvido repetida e exaustivamente.

As características não convencionais, especialmente no que diz respeito à dificuldade de formular matematicamente os problemas de scheduling, tem motivado diferentes trabalhos não só com respeito à modelagem, mas especialmente com respeito a estratégias eficientes para a busca de soluções, e sistemas capazes de gerenciar conflitos na demanda simultânea por recursos comuns.

CAPÍTULO 3 SEQUENCIAMENTO E SCHEDULING EM FLOW SHOPS
O Planejamento de Curto Prazo

3.1 Introdução

Neste capítulo tem-se como objetivos:

a) Apresentar e discutir as principais restrições que afetam o problema de scheduling;
b) Discutir as abordagens utilizadas para a incorporação das restrições às técnicas clássicas;
c) Apresentar as abordagens que levam à proposta do algoritmo apresentada no Capítulo 4.

O objetivo não é uma revisão bibliográfica exaustiva, mas analisar, para o caso do flow shop com restrições sobre recursos comuns, quais são as principais dificuldades associadas à definição de uma abordagem satisfatória para a solução do problema. Como resultado deseja-se reunir subsídios para fundamentar a abordagem proposta no Capítulo 4 para o scheduling em flow shops com restrições sobre os recursos comuns.

Para problemas de dimensão grande (N (número de tarefas) ≥10) as técnicas existentes geram, de uma forma geral, uma solução do problema em duas fases: na primeira não são considerados os recursos compartilhados, a exceção obviamente dos processadores, na segunda, mantendo inalterada a seqüência de execução das tarefas, os instantes de tempo das alocações são modificados de forma a que a demanda de recursos compartilhados seja compatibilizada com a oferta, ou seja, seja factível. Esta modificação dos instantes de alocação implica em aumento do tempo total para processamento das tarefas. O problema básico desta abordagem é que, quando a oferta de recursos compartilhados é crítica frente à demanda, o aumento total do tempo de processamento das tarefas pode ser grande, podendo existir outras seqüências de execução das tarefas factíveis com um tempo total de processamento menor.

Esta situação tem levado a propostas (Fox,1987) que sugerem uma mudança no enfoque da resolução do problema de scheduling nestas situações: procurar a solução através de uma metodologia orientada para a satisfação das restrições (Constrained Based Search) em lugar da abordagem clássica de minimização de um critério temporal de desempenho do sistema. A técnica proposta no Capítulo 4 é um passo nesta direção.

3.2. A importância das restrições na Programação da Produção em flow shops

O objetivo da Programação da Produção é a geração da distribuição temporal dos eventos significativos tais como início e término da ocupação dos processadores, início e término da preparação dos processadores e início e término das transferências de produtos entre processadores adjacentes, definidos de forma a garantir a factibilidade dos perfis de utilização dos processadores, armazenagem intermediária e recursos compartilhados, minimizando uma função de custo.

Esta distribuição temporal é normalmente apresentada na sua forma

gráfica (Carta de Gantt), e sua construção envolve decisões de seqüenciamento de tarefas e decisões de alocação.

A solução do problema de programação da produção do curto prazo deve satisfazer restrições que podem ser classificadas em três tipos:

- Restrições originadas pelo processo, como por exemplo restrições de precedência entre as diferentes tarefas;
- Restrições no prazo de entrega;
- Restrições originadas pela operação da planta, via de regra na forma de restrições sobre o volume de recursos compartilhados disponíveis.

As técnicas de programação da produção levam em consideração, simultâneamente ou não, os dois primeiros tipos de restrições. Já no que se refere a recursos comuns, as técnicas existentes apenas consideram os processadores, que são certamente recursos comuns com oferta limitada que não podem ser ignorados, mas não abordam outros recursos compartilhados. Na indústria química dois recursos tem importância especial: a armazenagem intermediária e alguns tipos de utilidades.

Para o objetivo deste trabalho duas características das restrições são especialmente importantes:

- A possibilidade ou não de relaxamento das restrições;
- A influência das restrições no seqüenciamento de tarefas e/ou na alocação das tarefas.

As duas características citadas acima fazem com que as restrições possam ter diferentes graus de importância relativa no desenvolvimento de uma programação. Com respeito à importância relativa as restrições podem ser classificadas em duas categorias básicas:

- Restrições de infactibilidade ou restrições "rígidas";
- Restrições flexíveis.

As restrições de infactibilidade ou rígidas são aquelas que infactibilizam um seqüenciamento de tarefas e/ou alocação de recursos e não aceitam relaxamento. São exemplos de restrições de infactibilidade :

- Restrições tecnológicas tais como restrições de precedência que infactibilizam determinadas sequencias de tarefas.
- Restrições em armazenagem intermediária, que infactibizam a alocacao do recurso, isto é, infactibilizam o acontecimento de um ou mais eventos simultâneos num certo instante de tempo.

As chamadas restrições flexíveis são aqueles que podem ser "relaxadas" pois admitem certa folga e/ou a sua violação pode ser aceita implicando em um aumento do custo total. Algumas restrições que podem ser tratadas como flexíveis são:

- Restrições no prazo de entrega;
- Restrições em recursos compartilhados, tais como mão-de-obra, energia elétrica etc.

Com respeito à forma como as restrições interferem no processo de resolução, elas podem ser classificadas em:

- Restrições "estáticas" : São restrições de infactibilidade que reduzem a priori o espaço de soluções factíveis, e nesta categoria são enquadradas as restrições de precedência;
- Restrições "dinâmicas" : São restrições cujo conflito é gerado no instante em que deve ser tomada uma decisão de alocação, tais como demanda simultânea de um recurso por duas ou mais tarefas. O conflito deve ser administrado da melhor maneira possível, quer seja através do relaxamento da restrição (se isto for aceitável), quer seja através de outras alternativas de decisão de alocação.

As restrições que infactibilizam a ocorrência simultânea de eventos tem despertado grande interesse na literatura e são o objetivo central deste trabalho. A seguir serão discutidos os problemas associados a cada uma das restrições mais frequentes nos problemas de scheduling em flow shops e que são:

- Armazenagem intermediária limitada;
- Recursos compartilhados.

3.2.1. A armazenagem intermediária

Tradicionalmente os processos químicos possuem armazenagem intermediária zero ou limitada dada a natureza física dos produtos processados na indústria química (normalmente gases, líquidos e suspensões), no entanto a armazenagem intermediária tem adquirido importância crescente nas mais variadas indústrias. (Leinstein (1990)).

O objetivo principal da armazenagem intermediária é oferecer maior grau de liberdade de ação, eliminando a necessidade de sincronismo entre processadores (Conway et al,1988). O custo mais importante associado à armazenagem intermediária é o "flow time" ou makespan.

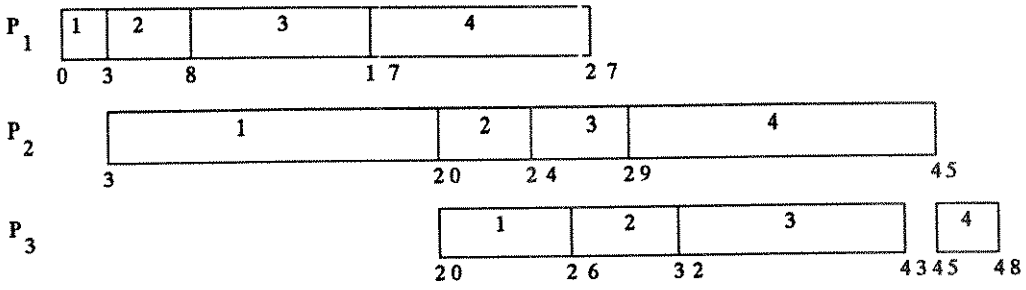
A disponibilidade ou não de armazenagem intermediária num determinado instante de tempo e, em caso positivo, a decisão sobre a sua utilização não é normalmente uma decisão a ser tomada com base em informações relativas exclusivamente a este instante de tempo em particular. Esta decisão depende de decisões em instantes anteriores (por exemplo, início de processamento de uma tarefa em um processador) e afeta decisões em instantes futuros (por exemplo, o grau de liberdade de decidir o início do processamento no próximo processador).

O maior grau de liberdade introduzido pela armazenagem intermediária não é utilizado via de regra na solução do problema da programação da produção, servindo como um meio para melhorar ou factibilizar a seqüência final obtida quando existem períodos ociosos ou problemas com a demanda de recursos compartilhados. É sabido porém que a utilização de armazenagem intermediária pode aumentar a produtividade (Conway et al,1988).

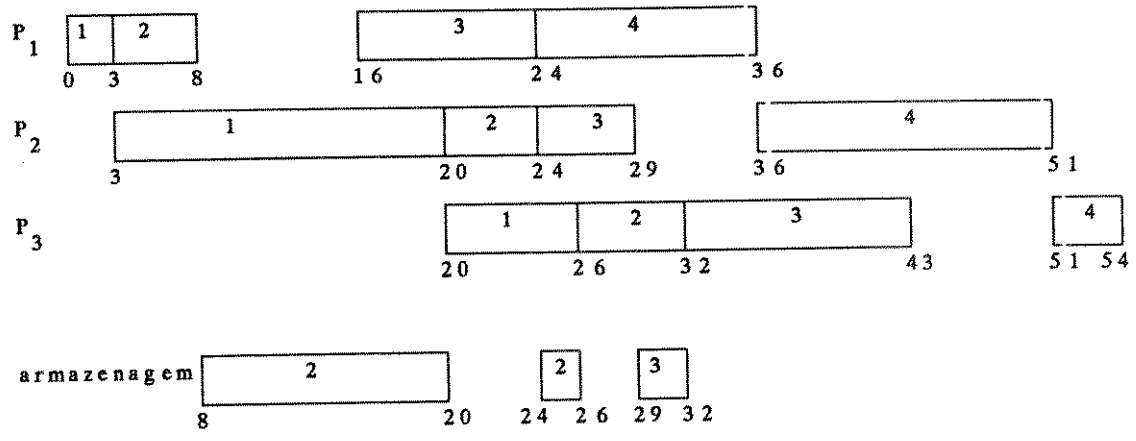
Os diferentes arranjos de armazenagem intermediária bem como as diferentes políticas de processamento geram diferentes categorias de problemas de Programação da Produção (PP). Por exemplo, para o caso do flow shop com armazenagem intermediária infinita -UIS, o problema de PP se reduz a um problema de seqüenciamento pois não são necessárias decisões de alocação da armazenagem. Neste caso a armazenagem existe em quantidade suficiente para acomodar qualquer perfil de demanda e o conhecimento da

seqüência de produção permite construir a Carta de Gantt. No caso da armazenagem intermediária ser limitada (flow shop FIS ou MIS) estão também envolvidas decisões de alocação e, como conseqüência não é possível a construção da carta de Gantt sem que, associadas às informações sobre a seqüência de produção, sejam também fornecidas as decisões acerca da alocação da armazenagem quando esta for solicitada por mais de uma tarefa simultâneamente.

Pode-se visualizar a diferença entre estes dois casos bem como o interesse em se dispor de armazenagem intermediária na Figura 3.1 onde estão apresentadas as Cartas de Gantt para uma mesma seqüência de tarefas em duas condições de processamento diferentes. No caso a) trata-se do flow shop UIS, e no caso b) dispõe-se de apenas um tanque de armazenagem compartilhado entre os estágios configurando um problema de limitação em armazenagem intermediária.



a) Carta de Gantt para o flow shop UIS



b) Carta de Gantt para o flow shop com um tanque de armazenagem intermediária compartilhado entre estágios

Figura 3.1 - Cartas de Gantt para o flow shop

Nos casos em que são necessárias decisões de alocação põem-se imediatamente dois problemas:

a) Quais estratégias de alocação da armazenagem devem ser empregadas no caso deste recurso ser solicitado por mais de uma tarefa ou produto simultâneamente;

b) Como expressar matematicamente os instantes de tempo de início e

término das tarefas.

O primeiro problema é o problema clássico do scheduling, isto é, decidir quando o recurso deve ser utilizado de forma mais satisfatória.O segundo é decorrente do primeiro e pode ser resolvido na medida em que sejam desenvolvidas fórmulas de recorrência para o cálculo do "completion time", que representa o instante em que o processador é liberado.No entanto estas fórmulas de recorrência dependem das decisões de seqüenciamento de eventos em instantes anteriores, sendo então impossível desvincular o cálculo do "completion time" das estratégias de alocação. A dificuldade em gerar expressões matemáticas para as restrições nos "completion time" para cada uma das abordagens clássicas para a solução dos problemas de Programação da Produção em flow shops com armazenagem intermediária finita estão discutidas no Anexo A.

3.2.2. Recursos compartilhados

Os recursos compartilhados tais como energia elétrica,vapor, água de refrigeração etc, estão disponíveis em quantidades limitadas, e as demandas individuais de cada tarefa são conhecidas. A relação matemática que descreve a condição de limitação nos recursos compartilhados será:

$$\sum_{i \in K} \sum_{j=1}^{M} Q^{l}_{ij}(t) \leq q^{l}(t) \qquad (3.1)$$

onde:
$Q^{l}_{ij}(t)$= consumo temporal do recurso l pela tarefa i no processador j no instante t

$q^{l}(t)$ = disponibilidade temporal do recurso l no instante t
K = conjunto de tarefas que demandam o recurso l no instante t

A equação (3.1) tal como apresentada mostra que, em cada instante a soma dos recursos solicitados para a execução das tarefas não pode ultrapassar o limite disponível de recursos naquele mesmo instante. Para se conhecer a demanda instantânea é necessário conhecer, tal como no caso da armazenagem intermediária, os instantes de ocorrência de cada evento o que significa conhecer os "completion time" (Anexo A). Assim conclui-se que, de fato, os dois tipos de restrição, embora apresentados como problemas diferentes na literatura, possuem graus de complexidade semelhantes. Isto significa que a solução deste tipo de problema exige, em princípio, que as decisões de seqüenciamento e alocação sejam tomadas simultâneamente, tal como no caso da armazenagem intermediária.

A principal diferença entre estes dois tipos de recursos compartilhados, e que conduz a tratamentos diferenciados na literatura, reside no fato que aos recursos compartilhados ditos "comuns" tais como energia elétrica, vapor, água de refrigeração etc, possuem perfis de demanda especificados pela receita tecnológica de execução da tarefa. No caso da armazenagem intermediária, a sua demanda é gerada por necessidades surgidas no decorrer da Programação da Produção.

3.2.3. Outras restricmes

As demais restrições normalmente presentes no scheduling em flow

shops são restrições do tipo estáticas, tais como as restrições de precedência e as restrições no prazo de entrega. Estas restrições interferem na geração do scheduling de maneira diferente das restrições citadas anteriormente.

Estas restrições não são menos importantes nem tampouco de solução mais simples que as demais restrições.Muito embora os problemas envolvendo tais restrições continuem sendo estudados em busca de abordagens satisfatórias, eles serão citados brevemente aqui por não constituirem o objeto central deste trabalho.

As restrições de precedência tem como principal consequência a redução do espaço de soluções na medida que os ordenamentos de tarefas ou operações que violem a restrição em questão são proibidos.

Os Problemas de Programação da Produção tem normalmente especificado um "prazo de entrega", no sentido que se tem normalmente definido um horizonte de tempo em que a programação deve ser desenvolvida. Além da definição de um horizonte de tempo, pode-se ter definidos prazos de entrega diferentes para cada produto.Normalmente a violação dos prazos de entrega implicam no aumento de custos financeiros.

Estas duas categorias de restrições embora semelhantes do ponto de vista formal, pois atuam explicitamente no ordenamento das tarefas, tem características diferentes com respeito à "flexibilidade". As restrições de precedência não podem ser violadas já que são restrições tecnológicas, enquanto que as restrições no prazo de entrega podem ser eventualmente relaxadas implicando na cobrança de uma penalidade.

3.3. Caracterização das Metodologias para a Solução dos Problemas de Scheduling

Nos problemas de Programação da Produção em flow shops deseja-se seqüênciar e alocar N tarefas em M processadores. Consequentemente existem, em princípio, $N!^M$ seqüências possíveis de produção, já que, em cada processador, podem ser estabelecidas N! seqüências diferentes (Baker (1973)).

Para o caso da indústria química interessam as chamadas seqüências de permutação, isto é, aquelas seqüências em que a ordem de processamento das operações de cada tarefa é a mesma em todos os processadores. A limitação do espaço de soluções às seqüências de permutação reduz o número de seqüências possíveis a N!.

Mesmo nos casos em que o espaço de soluções é limitado às seqüências de permutação, os problemas de Programação da Produção em flow shops são da classe de problemas NP (Não Polinomiais) (Garey et al. (1976)). Uma das características destes problemas é que o tempo necessário para resolvê-lo aumenta exponencialmente com a dimensão do problema tal como mostrado na Tabela 3.1. Nesta tabela são apresentados dados comparativos do tempo necessário para resolver problemas de diferentes dimensões usando algoritmos polinomiais e algoritmos de enumeração completa. Estes dados demonstram ser impossível pesquisar todas as soluções possíveis mesmo para problemas de pequena dimensão, sendo necessário recorrer a abordagens que permitam, de alguma maneira abreviar a busca de uma solução.

Deve-se ressaltar que, para cada seqüência possível de ser construída, são possíveis diferentes distribuições temporais dos eventos. A cada programação diferente de uma mesma seqüência de tarefas é associado um custo, aumentando ainda mais o número de soluções possíveis.

Tabela 3.1 - Dados comparativos de tempo de computação

N	10	15	20	40
ln N	1s	1.8s	2.6s	6.4s
N^2	1s	2.25s	4s	16s
N!	0.1s	10h	2000 anos	-

Formalmente o problema pode ser visualizado em dois níveis:

- Nível 1 : Seqüenciamento de tarefas;
- Nível 2 : Alocação de recursos,

que devem então ser resolvidos simultâneamente.

A solução eficiente do problema depende da escolha adequada do critério de otimização e do método de resolução.

A revisão da literatura permite distinguir duas categorias de problemas de Programação da Produção em flow shops:

- Seqüenciamento de tarefas;
- Scheduling de tarefas.

A classificação do problema em uma das categorias acima depende da natureza das restrições.

Um problema de programação da produção se classifica como um problema de seqüenciamento quando:

- A oferta de recursos comuns é ilimitada;
- A política de processamento e do tipo UIS, NIS ou ZW;
- Em cada estágio de processamento existe disponível apenas um processador.
- Não existe restrição no prazo de entrega das tarefas.

Neste tipo de problema o objetivo é determinar então a seqüência de processamento que otimize o critério de desempenho adotado, não sendo necessária qualquer decisão de alocação.

Em problemas de scheduling estão presentes restrições que exigem decisões acerca do instante de tempo factível em que as operações de cada tarefa devem e podem ser realizadas. As restrições que exigem tal tipo de decisão são a armazenagem intermediária limitada, recursos comuns e prazo de entrega.

Em resumo, a solução de um problema de seqüenciamento é constituída

apenas pela seqüência de produção. O conhecimento da seqüência de produção e da política de processamento (UIS, NIS, ZW) permite construir a carta temporal de eventos (Carta de Gantt) sem que seja necessária qualquer informação sobre a alocação dos processadores, já que não há qualquer limitação na oferta de outros recursos.

No caso de um problema de scheduling, a construção da carta temporal de eventos não pode ser feita com base exclusivamente no conhecimento da seqüência de produção das tarefas, já que os instantes de início das operações de cada tarefa estarão condicionados não só à disponibilidade dos processadores, mas também dependerão da capacidade do perfil de oferta dos recursos comuns acomodar uma nova demanda (Seção 4.2).

Em função das características gerais das restrições apresentadas, é desejável que a abordagem adotada para a solução do problema de Programação da Produção apresente flexibilidade para incorporar alternativas tais como (Fox (1990), Hughes e Smith (1985)):

- Capacidade de gerenciamento de conflitos gerados pelas restrições "dinâmicas". Esta é uma característica muito importante pois, como se verá (Secção 3.5) a dimensão e complexidade de alguns problemas de Programação da Produção impedem que sejam resolvidos satisfazendo todas as restrições simultâneamente. Nestes casos torna-se necessário recorrer ao alguma forma de hierarquização das restrições do problema, e no caso de as alternativas de gerenciamento do problema se esgotarem, permitir a interferencia externa do especialista.
- Recurso a ferramentas matemáticas adequadas e eficientes quando a formulação do problema e/ou o relaxamento de restrições permitirem a sua solução parcial ou integral.

As características apontadas acima têm por objetivo guiar a implementação de um sistema interativo, capaz de substituir o "human scheduler" nas suas atividades mais tediosas e repetitivas. A finalidade não é simular o raciocínio humano na confecção de schedules, até porque, por se tratar de uma atividade tediosa e repetitiva, é mais suscetível a existência de receitas desenvolvidas sem a adoção de parâmetros claros de otimalidade. É então desejável o desenvolvimento sistemas capazes de fornecer subsídios para a elaboração de schedules construídos a partir da captura do conhecimento do especialista no gerenciamento de recursos, e quando for possível o seu refinamento (Hughes (1985)).

Sacerdoti (1974) discute a dificuldade de se resolver de maneira eficiente os problemas de natureza combinatória e a importância da redução a priori do espaço de soluções. Para tanto ele propõe uma abordagem em que, usando uma representação simplificada do problema na qual detalhes facilmente incorporáveis posteriormente e não críticos para a construção do schedule sejam temporariamente ignorados, seja gerada uma solução ou conjunto de soluções que serão posteriormente exploradas em outros níveis de detalhamento. Uma vez obtida pelo menos uma solução, os detalhes são progressivamente adicionados de tal forma que no final resolve-se um conjunto de subproblemas do espaço original.

Este é um aspecto muito importante a ser considerado no scheduling de tarefas pois o número elevado de detalhes a serem considerados implicam que sejam empregados modelos e sistemas capazes de manipular todo este conjunto de detalhes. A qualidade da hierarquização dos subproblemas depende não só do problema global mas também da capacidade do especialista

em definir adequadamente os contornos de cada subproblema. Um exemplo de proposta de um sistema capaz de construir schedules em job shops na presença de restrições de importância relativa e natureza diferenciadas foi formulado por Fox (1983). O autor propõe um sistema de relaxamento seletivo das restrições segundo classificações definidas a priori, e um sistema que pontua a solução parcial ou a rejeita, caso a restrição não possa ser violada.

O objetivo neste trabalho é desenvolver e apresentar uma abordagem para o tratamento do problema do scheduling na presença de restrições dinâmicas que possa ser empregada em sistemas interativos de construção de schedules. Portanto, é importante discutir em linhas gerais quais as abordagens normalmente empregadas nas soluções dos problemas de scheduling, para identificar quais delas apresentam as características desejáveis que permitam sua inserção em sistemas globais de scheduling em flow shops.

Existem basicamente três tipos de abordagens que são utilizadas para a solução dos diferentes problemas de Programação da Produção. A escolha da abordagem adequada dependerá de fatores tais como:

1. Viabilidade de formulação matemática do problema;
2. Conhecimento de informações que possam abreviar a busca da solução.

As características básicas destas abordagens serão apresentadas a seguir, sendo apresentandas igualmente as vantagens e desvantagens normalmente associadas a cada uma delas.

3.3.1. Métodos de busca

A enumeração explícita de todas as soluções do problema é impraticável face à explosão combinatória; a alternativa é então recorrer a métodos de busca controlada para minimizar uma função de custo dada. Estes métodos funcionam através da aplicação de duas atividades : a) busca, que se concentra nas maneiras possíveis de exploração da estrutura em árvore e b) controle, que se concentra em selecionar, dentre as várias alternativas de caminhos que podem ser percorridos na busca da solução ótima, aquela que efetivamente será escolhida.

Os métodos de busca controlada permitem que as atividades de busca e controle sejam realizadas utilizando diferentes tipos de cálculo, bem como a incorporação de informações acerca da provável evolução do critério de otimização empregado, auxiliando na escolha do caminho mais eficiente na busca da solução ótima, e permitindo que sejam descartadas, ao menos temporariamente, soluções ineficientes (Nilsson (1971), Winston (1979), Dechter (1990), Kwa (1989), Berliner (1979)).

A solução é construída pelo método, no sentido que a ordenação e alocação das tarefas, que integram os objetivos da Programação da Produção, são decididas consecutivamente em cada nó da árvore.

Os problemas de scheduling tem sido tradicionalmente resolvidos através do recurso a métodos do tipo Branch and Bound(BAB).A utilização destes métodos na Programação da Produção especialmente nos problemas de seqüenciamento de tarefas, consiste em decidir sucessivamente qual a próxima operação a ser executada e quando executá-la, com base em

heurísticas para estimar o valor do critério de otimização. Em função da utilização praticamente unânime dos métodos tipo BAB para o seqüenciamento em flow shops, estes serão discutidos com mais detalhes nesta secção.

A estrutura do tipo árvore é composta de nós, que representam soluções parciais do problema, e ramos , que ligam nós de níveis sucessivos. Durante o processo de busca se terá na árvore diversos caminhos incompletos e um custo total associado a cada nó. O caminho ótimo é encontrado através da ramificação do nó com menor custo, gerando nós em um nível de profundidade da árvore imediatamente posterior. Este procedimento é aplicado ciclicamente até que a solução final é obtida.

A utilização eficiente de um método deste tipo apresenta uma grande limitação: o cálculo da função de custo associado a cada nó. Esta função custo é normalmente composta de duas parcelas:

- A primeira relativa ao custo associado à solução parcial representada no nó;
- A segunda relativa a uma estimativa de custo mínimo remanescente necessário para completar a solução parcial do nó.

A primeira parcela é fácil de ser calculada a partir dos dados do problema. A segunda parcela é mais complexa e implica na utilização de funções heurísticas para o cálculo de um custo mínimo associado ao resto do caminho em direção à solução final. É necessário garantir que o custo total resultante da soma destas duas parcelas não irá, em qualquer circunstância, ser reduzido quando se prossiga na árvore neste caminho. Se isto ocorrer, é invalidada toda estratégia de caminho ótimo proposta no método BAB.

É ainda desejável que a segunda parcela seja calculada com base em uma heurística bastante fiel ao modelo do problema pois, desta forma, o custo estimado se aproxima bastante do custo real. Como resultado é reduzido o número de ciclos necessários à construção da solução ótima, em outras palavras, é reduzida a freqüência do "backtracking". É claro que neste ponto deve ser respeitado um compromisso entre a frequência de backtracking e a complexidade da heurística empregada. Não é desejável que a heurística empregada se aproxime demasiadamente de uma "simulação" do caminho a ser ainda percorrido a partir do nó pois, desta forma, o custo computacional (tempo) necessário para calcular esta parcela será igualmente elevado aproximando a busca de uma enumeração completa.

Ao custo mínimo calculado com base na soma das duas parcelas citadas dá-se o nome limitante inferior ("lower bound") no qual é baseado, por sua vez, o controle da enumeração. Os métodos do tipo BAB apresentam uma série de características adequadas para a solução de problemas de scheduling em flow shops:

- É fácil a preservação do fluxo unidirecional das operações dentro do shop, não sendo necessário explicitar para tanto restrições de precedência das operações que compõem as tarefas naqueles problemas em que apenas as seqüências de permutação constituem o espaço de soluções. Neste caso basta decidir qual será a próxima tarefa a ocupar o shop.
- A busca pode ser limitada, isto é, podem ser definidas quais partes da árvore em que se deseja realizar a busca.
- Soluções parciais infactíveis podem ser aceitas pelo usuário através, por exemplo, do relaxamento seletivo de restrições.

Estas características mostram que, por sua estrutura, este é um método que favorece em princípio a incorporação de diferentes tipos de conhecimento que possam abreviar a busca de soluções, não exigindo para tanto, uma formulação matemática rigorosa do problema. Mas deve-se ressaltar que a sua eficiência está intimamente relacionada à qualidade do limitante inferior. Esta eficiência pode ser aumentada se for possível construir também um limitante superior, permitindo eventualmente a eliminação de caminhos que possam conduzir a soluções de pior qualidade que serão posteriormente descartadas (Baker (1974)).

3.3.2. Programação Matemática

A formulação matemática de um problema de scheduling como um problema de otimização é desenvolvida através de um procedimento do seguinte tipo:

- Análise do processo definindo variáveis e características específicas de interesse;
- Determinação do critério de otimização.
- Desenvolvimento do modelo de processo, de forma a definir as relações matemáticas entre as diversas variáveis envolvidas.

Dada a natureza do problema de Programação da Produção em flow shops, a aplicação deste procedimento resulta normalmente em um problema do tipo MILP (Mixed Integer Linear Problem) ou MINLP (mixed Integer Non Linear Problem) que pode ser resolvido através do emprego de métodos do tipo Decomposição de Benders (Geoffrion (1972)).

A formulação matemática do problema apresenta as seguintes limitações:

- Necessidade de formular matematicamente o modelo do processo;
- Uma vez estabelecido o modelo matemático do processo, este não pode ser alterado durante o processo de solução;
- O número de variáveis manipuladas é, em geral, bastante elevado, principalmente o número de variáveis binárias 0/1.

Como resultado destas características pode-se concluir que para os problemas cuja formulação matemática seja difícil de ser desenvolvida, esta não é obviamente a abordagem indicada. Além disso, mesmo quando o problema é possível de ser formulado, não há garantias de existir uma solução factível. Neste caso, a factibilização pode ser obtida pela modificação das equações do problema, ou pelo relaxamento das restrições. Isto significa que, a cada alteração proposta no modelo original, o problema deve ser resolvido integralmente de novo. Além disso, a resolução destes problemas não é fácil, especialmente pela presença de restrições de integralidade, que exigem também o recurso a técnicas de enumeração.

Para os problemas de scheduling este tem sido um caminho pouco explorado principalmente porque:

- Algumas restrições são difíceis de serem formuladas matematicamente;
- O número de variáveis é elevado, especialmente nos casos em que há compartilhamento de recursos.

3.3.3. Regras e Algoritmos Heurísticos

Neste tipo de abordagem é utilizado um conjunto de regras heurísticas operando sobre os dados ou sobre grandezas obtidas dos dados para gerar a seqüência de tarefas. A utilização desta abordagem é restrita aos problemas de seqüenciamento.

Não são utilizados procedimentos ou algoritmos de minimização de uma função de custo, mas apenas um conjunto de regras heurísticas que permitam decidir qual a próxima tarefa a integrar uma seqüência parcial.

As regras heurísticas poderiam ser obtidas através de dois caminhos:

- Resolução exaustiva do modelo do processo ;
- Observação do comportamento de sistemas reais.

No primeiro caminho, é necessário desenvolver o modelo do processo, resovê-lo exaustivamente e observar se as soluções obtidas obedecem um padrão dentro de limites aceitáveis.

No segundo caminho as regras heurísticas são estabelecidas como consequência da observação que, sob determinadas circunstâncias ou condições operatórias, o comportamento do processo obedece padrões que podem ser descritos através de regras ou procedimentos simplificados.

Na literatura existem algumas propostas de regras heurísticas que traduzem atitudes "razoáveis" para o seqüenciamento e alocação de recursos (Seção 3.5.1.3) mas a sua eficiência num problema particular não pode ser estabelecida a priori.

3.3.4. Comentários

É importante ressaltar que a escolha da metodologia a ser empregada na solução do problema deve apresentar pelo menos duas características básicas: a) flexibilidade para acomodar variações significativas nos níveis de tolerância na oferta de recursos compartilhados nos perfis de demanda de produtos/tarefas e na política de processamento e b) flexibilidade para acomodar com relativa facilidade heurísticas de decisão inferidas a partir da observação do problema real e interferências externas no processo decisório. Estas características desejáveis apontam para os métodos de busca como sendo aqueles com estrutura flexível o suficiente para satisfazer estas exigências.

A experiência acumulada no scheduling de tarefas em outras áreas, tais como o scheduling em job shops, reforçam a escolha dos métodos de busca inseridos em sistemas hierárquicos de Programação da Produção, como os mais indicados. A inexistência de formulações matemáticas para problemas complexos de Programação da Produção (Secção 3.4.3.4) também apontam na direção da utilização dos métodos de busca (Fox (1983), Smith e Ow (1990)).

3.4. Revisão das técnicas existentes para o problema de sequenciamento em flow shops

Uma das categorias de problema de Programação da Produção mais

estudada é o seqüenciamento de tarefas em flow shops. Um problema de Programação da Produção pode ser classificado como um problema de seqüenciamento nas seguintes condições:

- Cada estágio do flow shop é composto de apenas um processador, não havendo portanto necessidade de alocar a tarefa a um processador específico;
- A política de processamento é do tipo UIS, NW ou NIS;
- A oferta em recursos compartilhados é ilimitada.

Além destas hipóteses, as técnicas e abordagens existentes limitam o espaço de soluções às seqüências de permutação.

As técnicas mais importantes de seqüenciamento em flow shops serão apresentadas, mas primeiramente serão definidas as principais variáveis e funções de custo utilizadas em problemas de seqüenciamento.

3.4.1. Definição de variáveis e funções de custo mais utilizadas

O conjunto de informações que deve ser definido na abordagem de um problema de seqüenciamento são (Baker (1973)):

t_{ij} = tempo de processamento da tarefa i no processador j, i=1,..N e j=1,..,M;

r_i = (ready time) instante em que a tarefa i está pronta para ser executada;

d_i = (due date) instante em que a tarefa i deve estar terminada.

O resultado do seqüenciamento será:

C_{ij} = (Completion time) instante em que a tarefa i está terminada no processador j , em particular C_{iM} representa o instante de tempo em que a tarefa i está pronta e pode deixar o shop;

que será obtida a partir da minimização de uma função custo baseada em variáveis tais como (Baker (1972)):

F_i = (Flow time ou tempo de residência) quantidade de tempo que uma tarefa permanece no sistema = $C_{iM} - r_i$;

L_i = (Lateness ou defasagem) quantidade de tempo que excede a data desejada para o término da tarefa = $d_i - r_i$;

T_i = (Tardiness ou atraso) definido como a defasagem positiva (atraso), quando $L_i > 0$, ou zero se a defasagem é negativa ($L_i \leq 0$)

Os critérios mais frequentemente empregados na otimização são a minimização de:

a)Tempo médio de residência(F^*)

$$F^* = (1/N) \sum_{i=1}^{N} F_i \qquad (3.2)$$

b)Atraso médio(T^*)

$$T^{*} = (1/N) \sum_{i=1}^{N} T_i \qquad (3.3)$$

c)Tempo de residência máximo(F_{max})

$$F_{max} = \max_{1 \le i \le N} \{F_i\} \qquad (3.4)$$

d)Atraso máximo(T_{max})

$$T_{max} = \max_{1 \le i \le N} \{T_i\} \qquad (3.5)$$

e)Número de tarefas atrasadas(N_t)

$$N_t = \sum_{i=1}^{N} N(T_i) \quad \text{onde} \quad \begin{cases} N(T_i)=1, & \text{se } T_i \ge 0 \\ N(T_i)=0, & \text{caso contrário} \end{cases} \qquad (3.6)$$

Todas estas medidas de desempenho do sistema são chamadas medidas "regulares" de desempenho, e são sempre funções do tempo de término da tarefa, isto é:

$$Z=f(C_{1M}, C_{2M}, \ldots\ldots, C_{NM})$$

Uma medida de desempenho é definida como "regular" se:

1)O objetivo do seqüenciamento é minimizar Z;
2)Z aumenta apenas se o tempo de término das tarefas também aumenta.

O problema se seqüenciamento não é simples de ser resolvido. Para se apreender em linhas gerais o grau crescente de complexidade do problema em função da dimensão (número de tarefas e processadores) e especialmente do tipo de critério adotado na otimização será apresentada inicialmente a estrutura mais simples de processamento que é o caso de um estágio/um processador.

3.4.2. Processamento em um estágio

O exemplo mais simples de processamento é o caso do processamento em um estágio e um processador. Este tipo de processamento é frequentemente utilizado para ilustrar o problema de seqüenciamento de tarefas e será aqui utilizado para introduzir algumas das idéias básicas que permeiam a problemática de seqüenciamento.

Neste caso deseja-se sequenciar N tarefas sendo que cada uma delas deve ser processada em um único estágio, e dispõe-se apenas de um processador. Este tipo de problema tem sido objeto de vários estudos de forma que são conhecidos resultados de caráter geral para vários tipos de especificação do problema.

A maior parte das soluções básicas foram desenvolvidas admitindo-se o seguinte conjunto de hipóteses:

H_1)r_i=0, 1≤i≤N, isto é, todas as tarefas estão prontas para o processamento em t=0, de forma que ficam automaticamente eliminadas as restrições de precedência;

H_2)O tempo de preparação do processador não depende da seqüência e está incluído no tempo de processamento;

H_3)O processamento de uma tarefa não pode ser interrompido;

H_4)O processador não está ocioso enquanto houver tarefa a ser executada.

Na Tabela 3.2 são apresentados alguns resultados existentes para diferentes funções de custo.

A utilização de outros critérios diferentes dos apresentados na Tabela 3.2 implica em soluções mais complexas do que as soluções apresentadas nesta tabela. A obtenção das soluções ótimas nestes casos necessitam, em geral, a utilização de ferramentas de otimização combinatória, tais como métodos do tipo branch-and-bound (BAB) e programação dinâmica. Uma alternativa é a utilização de regras sub-ótimas, tais como métodos heurísticos.

Tabela 3.2 - Resultados básicos para o caso n tarefas/1 estágio/1 processador (Baker (1973)

Critério de desempenho	Tipo de seqüência
a) Tempo médio de residência $F^{*}=(1/n)\sum_{j}^{N} F_j$	SPT(Shortest Processing Time) $t_{[1]}<t_{[2]}<t_{[3]}<..<t_{[N]}$
b) Tempo médio de residência ponderado $F=\frac{\sum_{i=1}^{N} w_i F_i}{\sum_{i=1}^{N} w_i}$	WSPT(Weighted Shortest Processing Time) $\frac{t_{[1]}}{w_{[1]}}<\frac{t_{[2]}}{w_{[2]}}<\frac{t_{[3]}}{w_{[3]}}<..<\frac{t_{[N]}}{w_{[N]}}$
c) Atraso médio $L^{*}=(1/N)\sum_{i=1}^{N} L_i$	SPT
d) Atraso máximo $L_{max}=\max(L_i)$ ou $T_{max}=\max(T_i)$	EDD(Earliest Due Date) $d_{[1]}<d_{[2]}<....<d_{[n]}$
Obs: [j] denota a ordem da tarefa na sequencia, por exemplo 5=[1], indica que a tarefa número 5 é a primeira tarefa da seqüência	

À medida que são levantadas cada uma das hipóteses H_1 a H_4 apresentadas anteriormente, o problema torna-se progressivamente (Potts (1974) mais complexo. Um caso frequente é aquele em que o tempo de processamento é dependente da seqüência, como por exemplo na indústria de tintas, e sua solução pode ser obtida por um algoritmo do tipo BAB ou através da formulação como um problema de programação inteira. Outra restrição frequente na indústria química é a restrição tecnológica de precedência. Tome-se como exemplo um conjunto de três tarefas (a,b,c) com $t_a<t_b<t_c$.A seqüência ótima no caso de se desejar minimizar o tempo médio de residência é, como se pode ver na Tabela 3.2, uma seqüência SPT ou seja, a seqüência (a-b-c). Se existir uma restrição de precedência do tipo c<a,ou seja c deve ser executado antes de a, não é evidente qual das seqüências remanescentes (c-a-b, b-c-a ou c-b-a), será ótima. Neste caso, uma alternativa pode ser a utilização de uma abordagem do tipo BAB.

3.4.2.2. N tarefas/ 1 estágio/ M processadores

O caso em que se tem (N tarefas/ 1 estágio/ M processadores), isto é, há disponibilidade de diferentes processadores no estágio capazes de executar a mesma operação deixa de ser um problema de seqüenciamento para se tornar um problema de scheduling já que é necessário tomar decisões de seqüenciamento e alocação de recursos através da decisão de qual processador deverá executar cada uma das tarefas. Os critérios de desempenho do sistema mais frequentemente empregados neste caso são o tempo de residência ponderado, tempo de residência máximo (Makespan) e atraso ponderado.

Este problema se reduz a um problema de seqüenciamento apenas no caso em que: a) todos os processadores são idênticos, e b) as tarefas podem ser suspensas durante o seu processamento. Nestas condições não é necessário alocar a tarefa a um processador já que todos são idênticos, além disso a possibilidade de suspender as tarefas durante sua execução, permite que as tarefas sejam parcialmente realizadas em diferentes processadores.

Um dos algoritmos básicos para o seqüenciamento de tarefas no caso (N/1/M) é o algoritmo de McNaughton (Baker 1973), para a minimização do makespan nas condições limite apontadas acima em que os processadores são idênticos e é permitida a preempção de tarefas. Neste caso o makespan mínimo (M^*) é dado por:

$$M^* = \max \left((1/M) \sum_{i=1}^{N} t_i, \max [t_i] \right) \qquad (3.7)$$

3.4.3. Estágios em serie (flow shop)
3.4.3.1. Características Gerais

A principal característica do flow shop é a sua relação de precedência entre os estágios. A estrutura é composta de M estágios consecutivos, e cada tarefa é formada por M operações. Cada uma das operações deve ser realizada em um só estágio obedecendo uma direção única de fluxo de operações. Em cada estágio pode haver um número variável de processadores capazes de executar operações idênticas, mas com capacidades e taxas de processamento que podem ser diferentes caracterizando um "network flowshop". O caso mais simples é aquele em que cada estágio é formado por apenas um processador caracterizando o flow shop puro, que é o objeto central deste trabalho. Neste caso estágio e processador se confundem sendo normalmente empregada a terminologia "processador".

Cada uma das tarefas, para ser completada, necessita que seja executado um conjunto de operações, e cada uma destas operações será executada em um processador diferente. Os processadores são numerados de 1 a M, e as operações da tarefa i são numeradas na forma (i,1),(i,2),.....,(i,M). Isto quer dizer que se a operação j precede a operação k então o processador alocado para desenvolver a operação j possui numeração inferior ao processador alocado para desenvolver a operação k (j<k).No caso mais geral são necessárias i operações para que uma tarefa seja executada (com i<M) e neste caso o tempo de processamento das (M-i) operações restantes é igual a zero.

Em relação ao caso citado anteriormente de um estágio/um processador, o problema apresenta algumas características novas, sendo que as principais são:

- Explosão combinatória do número de seqüências que forma o conjunto de soluções;
- Possível interesse na inserção de períodos ociosos na execução de tarefas, o que implica, mesmo no caso do flow shop puro, na necessidade de estabelecer uma política de armazenagem intermediária ou processamento.

As hipóteses normalmente adotadas são as seguintes:

H_1.N tarefas estão disponíveis para o processamento no tempo zero, e cada tarefa necessita M operações,cada uma delas em um processador diferente (flow shop puro);

H_2.Os tempos de preparação dos equipamentos são independentes da seqüência e estão incluídos nos tempos de processamento;

H_3.M processadores diferentes estão disponíveis continuamente;

H_4.Não é permitida a preempção de tarefas nos processadores

No caso de 1 estágio/1 processador existem N! seqüências possíveis de serem examinadas implicitamente ou explicitamente.No caso do flow shop existem N! seqüências possíveis em cada processador, o que conduz a $N!^M$ seqüências possíveis, isto porque a adoção das hipóteses H_1 a H_4 não impõe que as seqüências de execução das operações sejam as mesmas em todos os processadores.

Garey (1976) mostra que as seqüências de permutação constituem um conjunto dominante do espaço de soluções factíveis mas nem sempre contem a solução ótima. O espaço de soluções formado pelas seqüências de permutação contem comprovadamente a solucao otima em apenas dois casos (Baker (1973), Garey (1976)):

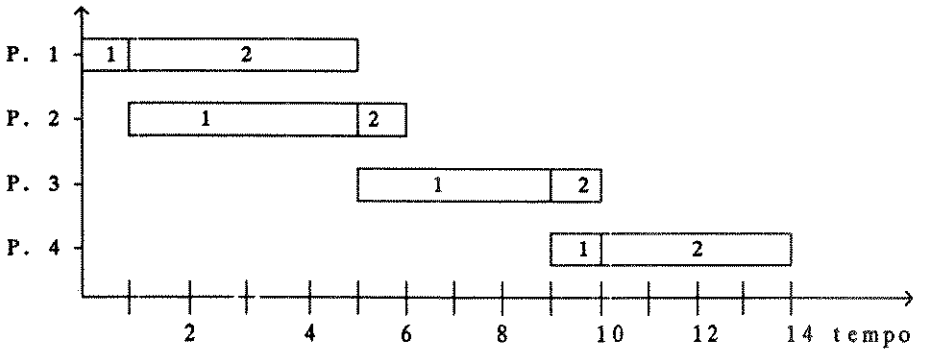
1. Otimização de qualquer medida regular de desempenho quando M=2.;
2. Otimização do makespan (tempo máximo de residência) quando M=2 ou M=3.

Isto quer dizer que, para problemas diferentes dos citados, a seqüência que irá otimizar o critério escolhido não será necessariamente uma seqüência de permutação. No entanto, as diferentes abordagens propostas na literatura restringem o espaço de soluções ao subconjunto de seqüências de permutação. No caso dos processos químicos a manutenção do ordenamento das tarefas em todos os equipamentos é de fato uma restrição tecnológica, de modo que a solução desejada será sempre uma seqüência de permutação.

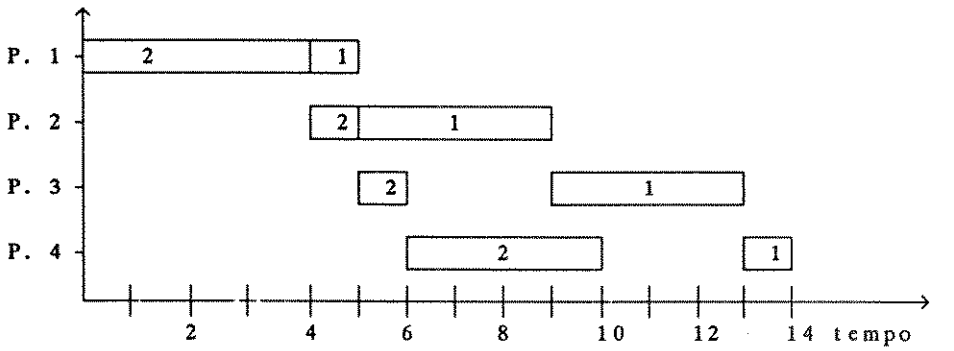
Na Figura 3.2 está ilustrada a situação em um flow shop em que a não limitação da solução às seqüências de permutação permite obter uma solução com melhor valor do tempo de residência médio ($\overline{F}$). Os dados correspondentes ao exemplo estão apresentados na Tabela 3.3.

Tabela 3.3 - Tempos de Processamento

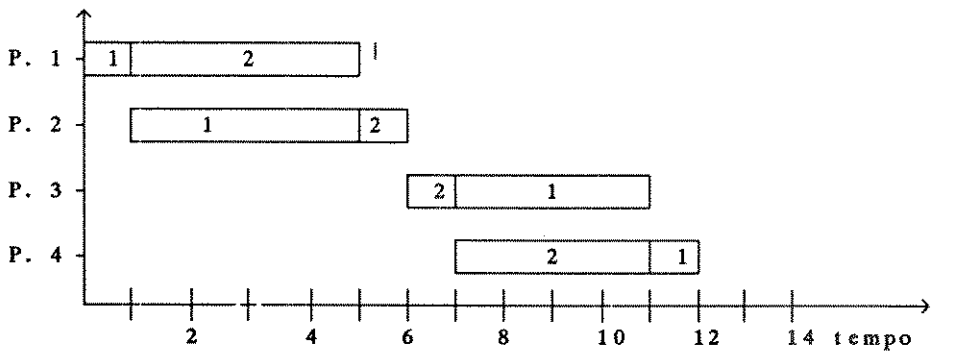
Processador	Tarefa 1	Tarefa 2
1	1	4
2	4	1
3	4	1
4	1	4



a) Tempo médio de residencia $\overline{F}$= 12



b) Tempo médio de residencia $\overline{F}$= 12



c) Tempo médio de residencia $\overline{F}$= 11,5

Figura 3.2 - Exemplo de inserção vantajosa de períodos inativos (Baker (1972))

Repare-se na Figura 3.2 que no caso c, em t=5, o processador 3 está disponível para iniciar o processamento da tarefa 1, no entanto é inserido um período inativo conduzindo a um tempo médio de residência inferior ao

dos casos anteriores onde não há períodos inativos.

No caso de se restringir a solução às seqüências de permutação, a inserção de períodos ociosos gerados pela alteração da ordem de processamento das operações não poderá ocorrer, e a vantagem de inserção de períodos ociosos devido a outras razões não é evidente. De qualquer modo, a inserção de períodos ociosos passa a estar fortemente associada à disponibilidade da armazenagem intermediária, ou à política de processamento. Não existe até o momento uma metodologia capaz de prever em qual situação a inserção de períodos ociosos possa ser vantajosa para a redução do tempo total de processamento de forma que este artifício, embora interessante, tem uma aplicação real limitada quando se considera exclusivamente a minimização do tempo total de processamento.

3.4.3.2. Algoritmo de Johnson

O Algoritmo de Johnson foi desenvolvido para o seqüenciamento de tarefas em um flow shop com apenas dois processadores, com armazenagem infinita entre os estágios e usando como critério de otimização o makespan. Johnson desenvolveu um algoritmo construtivo que conduz à solução ótima e é apresentado a seguir.

ALGORITMO DE JOHNSON

1.Determine $t^{*}_{ij} = \min_{j=1,2} \{t_{ij} : i=1,2,...N\}$

onde:

j=processador 1 ou 2;
i=número do produto ou tarefa;
t_{ij}=tempo de processamento da tarefa i no processador j

2.Se j=1, a tarefa i é seqüenciada tão cedo quanto possível e é retirada da lista de tarefas ainda não seqüenciadas;

3.Se j=2, a tarefa i é seqüenciada tão tarde quanto possível e é retirada da lista de tarefas ainda não seqüenciadas.

Exemplo de Aplicação

Tabela 3.4 - Exemplo de aplicação do Algoritmo de Johnson

Tarefa	Tempos de Processamento	
	Processador 1	Processador 2
1	5	9
2	3	6
3	8	2
4	6	2
5	1	4
6	7	5

Tarefa 5 seqüenciada	5 – – – – –
Tarefa 3 seqüenciada	5 – – – – 3
Tarefa 4 seqüenciada	5 – – – 4 3
Tarefa 2 seqüenciada	5 2 – – 4 3
Tarefa 1 seqüenciada	5 2 1 – 4 3
Tarefa 6 seqüenciada	5 2 1 6 4 3

A grande importância deste resultado reside no fato que, embora o seqüenciamento de tarefas seja um problema de natureza combinatória, este é um algoritmo polinomial(Johnson (1965)). Como conseqüência este algoritmo tem sido empregado para o desenvolvimento de heurísticas de seqüenciamento em flow shops UIS com um número de processadores superior a dois (Knopf (1982), Reklaitis (1985)), embora, como se verá mais adiante, estas heurísticas frequentemente não conduzirem a bons resultados.

A maior parte dos problemas de seqüenciamento em flow shops ocorre em sistemas com um número de processadores superior a dois conduzindo a problemas da classe NP (Garey et al. (1976)).

A explosão combinatória faz com que o tempo necessário para a obtenção da solução ótima cresça exponencialmente com a dimensão do problema. A inexistência de algoritmos construtivos, tal como o algoritmo de Johnson para problemas com número de processadores maior que dois faz com que sejam utilizadas outras abordagens para solucionar o problema de seqüenciamento em flow shops.

3.4.3.3. Técnicas de enumeração

Os problemas de seqüenciamento são candidatos naturais a serem resolvidos através do uso de métodos de enumeração em função da sua estrutura particular. Na Figura 3.3 é apresentada a estrutura em "árvore" resultante da enumeração de todas as soluções de um problema de seqüenciamento em um flow shop em que se deseja pesquisar apenas as seqüências de permutação. Diante da impossibilidade de enumerar todas as soluções do problema devido à explosão combinatória, uma das opções é recorrer a um esquema de enumeração inteligente ou controlada.

Os métodos de enumeração do tipo BAB são muito empregados no seqüenciamento de tarefas em flow shops, chegando muitas vezes a se confundir seqüenciamento de tarefas em flow shops e a abordagem BAB. Isto ocorre pois neste caso a direção do fluxo de processamento das tarefas é rígido e fácil de ser preservado em uma estrutura em "árvore", já que a tarefa só abandona o shop após o seu término.

A idéia deste método é, com base em uma função objetivo avaliada em cada nó da árvore, decidir qual deverá ser a próxima tarefa a integrar a seqüência parcial construída até o momento, isto é, a escolha dos ramos a serem percorridos é feita com base no valor da função objetivo dos diferentes nós candidatos.

Existem diferentes abordagens BAB e diferentes critérios que podem ser empregados no seqüenciamento de tarefas em um flow shop. Conforme apontam Baker(1975) e Nilsonn (1971), a escolha do tipo de método BAB (Depth First, Breadth First ou Best First) e da estratégia empregada na avaliação do critério a ser empregado na ramificação dos nós, é uma escolha "pessoal", na medida em que depende da familiaridade do usuário com o

método e com o problema, não existindo evidências de superioridade de nenhum dos métodos citados.

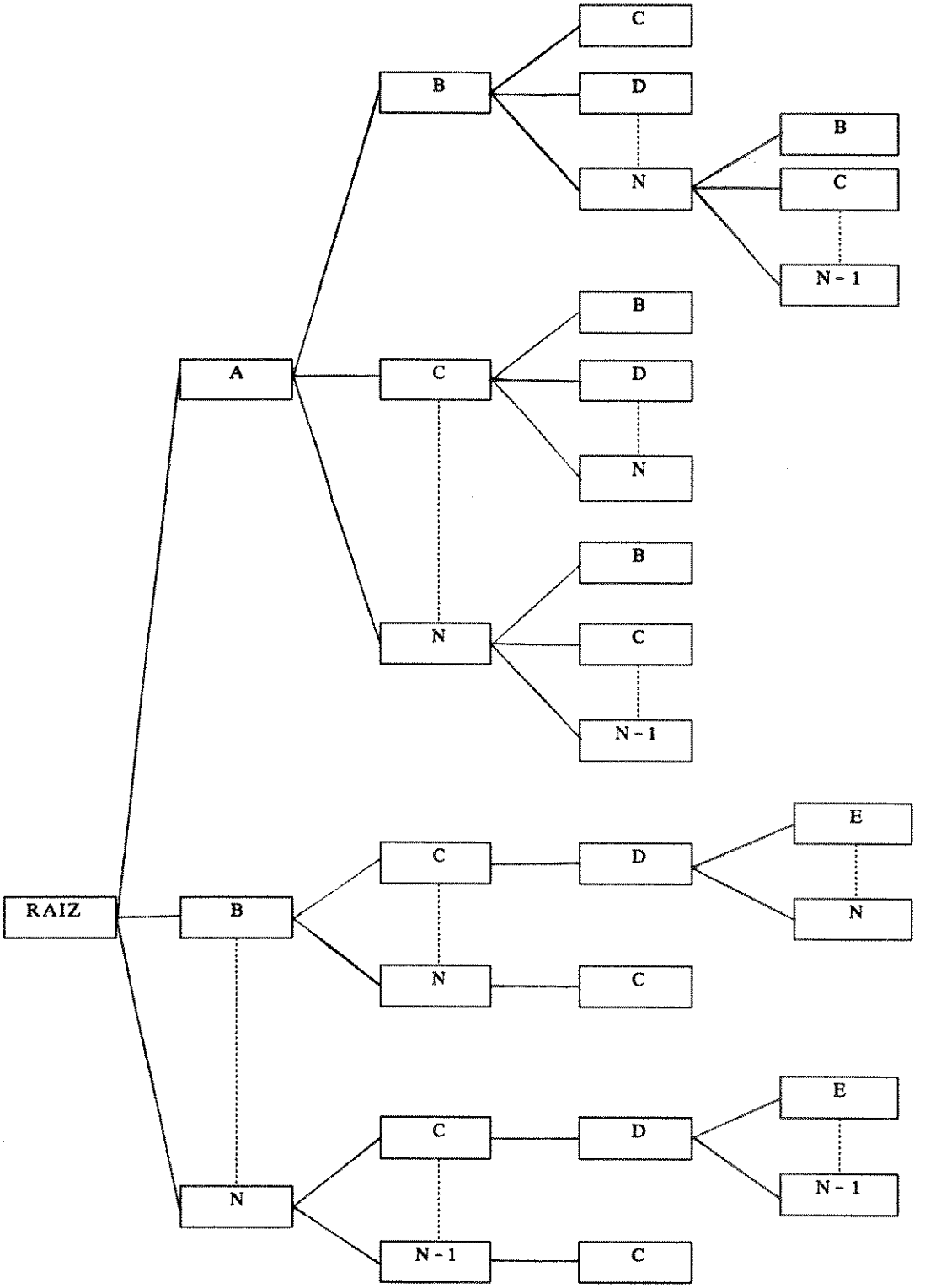


Figura 3.3 - Estrutura em árvore

A escolha, dentre as tarefas não seqüenciadas, da próxima tarefa a integrar a seqüência parcial é normalmente feita com base em estimativas do custo ou desempenho do sistema calculados com base nas duas parcelas já citadas (Seção 3.3.1): uma parcela referente ao custo ou desempenho do sistema associado ao conjunto de tarefas seqüenciadas e a outra parcela

referente à estimativa do custo ou desempenho do sistema associado às tarefas não seqüenciadas. Quanto melhor for a estimativa do comportamento futuro da seqüência, em outras palavras, quanto mais a heurística empregada se aproximar das hipóteses do modelo do problema em estudo, maior será a probabilidade de se escolher rapidamente o melhor caminho na busca de uma solução.

Um dos problemas desta natureza mais estudados é o seqüenciamento de tarefas em um flow shop adotando-se uma política UIS e adotando como critério de otimização a minimização do makespan. Neste caso ter-se-ão as seguintes parcelas:

– A primeira parcela representa o makespan da seqüência parcial calculado com base no instante de término das tarefas C(i,j), que reflete a distribuição temporal da tarefas já seqüenciadas, e portanto representa o tempo efetivamente comprometido na execução de uma determinada seqüência parcial. No caso mais simples em que o tempo de processamento já inclui os tempos de transferência e preparação do equipamento, este tempo pode ser calculado pela expressão (3.8).

$$C(i,j)=\max[\ C(i-1,j);C(i,j-1)\ ] + t_{ij} \qquad (3.8)$$

onde:

C(i,j)=instante de tempo em que a tarefa com posição i na seqüência é terminada no processador j

t_{ij} =tempo de processamento da tarefa i no processador j

- A segunda parcela representa uma estimativa do tempo ainda a ser gasto na execução das demais tarefas ainda não seqüenciadas. A soma destas duas parcelas resulta no limitante inferior, e alguns dos mais frequentemente usados são apresentados na Tabela 3.5. Cada um destes limitantes implica em adotar uma "estratégia" diferente para estimar o tempo mínimo necessário para processar as tarefas não seqüenciadas no nó em questão.

A seguir são apresentados alguns comentários das idéias envolvidas no desenvolvimento de cada uma das heurísticas apresentadas na Tabela 3.5.

No caso da heurística SMBB (Simple Machine Based Bound) o limitante inferior é construído através da adição do tempo necessário para processar as tarefas remanescentes no último processador.

O cálculo do limitante inferior em cada processador segundo a heurística FMBB (Full Machine Based Bound) é composta de três parcelas:

a) intervalo de tempo que o processador j já está comprometido;

b) o tempo de processamento ainda necessário no processador j;

c) o tempo mínimo de processamento ainda necessário no shop tendo em conta que o processamento em j já está terminado.

As tarefas não podem ocupar o mesmo processador ao mesmo tempo. A heurística NIB (Non Interference Bound) se baseia em resolver este conflito através da estimativa do instante potencial em que as tarefas podem ter seu processamento iniciado no último processador (C_i). Estes instantes são calculados e ordenados em ordem crescente, calculando-se em seguida o valor de v_i recursivamente, tomando-se $v_o = 0$.

Os cálculos do limitante da heurística JBB (Job Based Bound) se baseiam em identificar qual processador é, para uma determinada seqüência parcial, o gargalo do processamento.

Tabela 3.5 - Expressões para o cálculo de "lower bounds" para o flow shop UIS (Baker (1977))

1) Simple Machine Based Bound

$$LB = C(K,M) + \sum_{i \in U} t_{iM}$$

2) Full Machine Based Bound

$$B_j = C(K,j) + \sum_{i \in U} t_{ij} + \min_{i \in U} \left\{ \sum_{l=j+1}^{M} t_{il} \right\} \quad 1 \le j \le M-1$$

$$B_M = C(K,M) + \sum_{i \in U} t_{iM}$$

$$LB = \max_{1 \le j \le M} \{B_j\}$$

3) Non Interference Bound

$$C_i^* = C(K,1) + \sum_{j=1}^{M-1} t_{ij} \qquad i \in U$$

C_i^* = instante "potencial" em que a tarefa i pode ser iniciada no processador M. Estes instantes são ordenados em ordem crescente

$$v_i = \max \{C_i^*, v_{i-1}\} + t_{iM}, \quad \text{com } v_o = 0$$

v_i = instante mais cedo que a tarefa i poderia ser terminada no processador M

$$LB = \max_{i \in U} \{v_i\}$$

4) Job Based Bound

$$d_j = C(K,j) + \max_{i \in U} \left\{ \sum_{k=j}^{M} t_{ik} + \sum_{\substack{l \in U \\ l \ne i}} [t_{lj}, t_{lM}] \right\}$$

$$LB = \max_{1 \le j \le M-1} \{d_j\}$$

Dentre as várias heurísticas apresentadas na Tabela 3.5 a heurística FMBB (Full Machine Based Bound) é a mais frequentemente citada na literatura. Embora não haja qualquer evidência da superioridade do desempenho desta heurística frente as demais citadas nesta tabela, ela será utilizada neste trabalho sempre que for necessário estimar o makespan de flow shops na ausência de limitação na oferta de recursos compartilhados, principalmente nos algoritmos apresentados no Capítulo 4. No exemplo a seguir é mostrada a aplicação do BAB Best First ao seqüenciamento de

tarefas em um flow shop UIS.

Na Tabela 3.6 são apresentados os tempos de processamento de um exemplo e na Figura 3.4 a árvore de busca gerada a partir deste exemplo usando a heurística "Full Machine Based Bound" (FMBB).

Tabela 3.6 - Tempos de Processamento

Tarefa	Processador		
	1	2	3
A	3	4	10
B	11	1	5
C	7	9	7
D	10	12	2

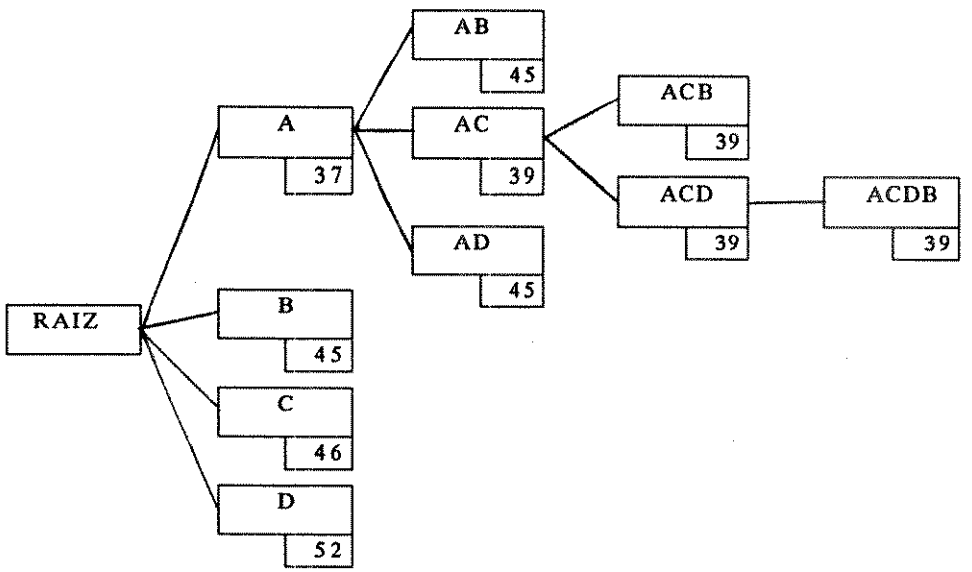


Figura 3.4 - Exemplo de aplicação de um método BAB Best First (Dados da Tabela 3.6)

A escolha do caminho a ser seguido na árvore de busca pode ser feita não apenas com base na estimativa de funções de custo mas também com base nas chamadas relações de dominância. Nesta estratégia, uma vez selecionado o nó a ser expandido, é desenvolvida uma atividade de sondagem baseada em relações de dominância para escolher os ramos que devem ser percorridos. A principal diferença desta estratégia com relação à estratégia de seqüenciamento usando o BAB da forma exposta anteriormente é que, no primeiro caso, uma vez escolhido o nó a ser expandido, todos os ramos derivados deste nó são igualmente candidatos e será calculado o valor estimado da função custo para cada tarefa candidata a integrar a seqüência parcial. No caso de se utilizar as relações de dominância, o objetivo é identificar quais ramos de um nó candidato são ineficientes. Isto equivale a dizer que, embora o nó candidato seja caracterizado por uma função custo, podem existir ramos gerados neste nó que certamente constituirão soluções dominadas, isto é, o custo associado a estes ramos serão certamente superiores aos custos dos demais ramos candidatos de um mesmo nó. Isto equivale a dizer que, se a partir de um nó candidato puderem ser gerados X ramos, e destes Y sejam ramos dominados identificados através das relações de dominância apropriadas, apenas (X-Y) ramos serão sondados. Na forma clássica do BAB todos os X ramos são igualmente candidatos e serão

igualmente sondados.

Para o caso do flow shop UIS são conhecidas as relações de dominância que permitem utilizar esta estratégia no seqüenciamentode tarefas(Baker (1975), Lageweg(1978)).

Baker (1975) mostra que, para o caso do seqüenciamento de tarefas em um flow shop UIS, a eficiência da busca, representada pelo número de nós sondados, pode ser aumentada pela combinação das duas estratégias citadas de controle da enumeração: limitante inferior e relações de dominância.

A utilização eficiente de métodos de busca na solução de problemas exige o cálculo de um limitante inferior com base em funções ou procedimentos de estimativa de custo adequadas e que reflita com propriedade o problema estudado. Uma boa heurística permite ao usuário rejeitar muitos dos caminhos prováveis de solução (Sacerdoti (1974). Além disso a eficiência é também resultado de uma solução de compromisso entre uma estimativa rigorosa do custo de cada caminho (ou solução parcial) e o número total de caminhos sondados.

O exemplo a seguir ilustra como a não utilização de uma heurística para a estimativa do custo associado a cada nó adequada ao tipo de problema aumenta a frequência do backtracking.

Seja um flow shop com M=3 processadores no qual se deseja processar N=4 tarefas, cujos tempos de processamento, tempos de transferência entre equipamentos estão apresentados nas Tabelas 3.7 e 8 respectivamente.

Tabela 3.7 : Tempos de Processamento

Processa-	Tarefa			
dor	1	2	3	4
1	4	4	5	2
2	6	6	5	5
3	3	4	9	8

Tabela 3.8 : Tempos de Transferência*

Processa-	Tarefa			
dor	1	2	3	4
0	2	1	1	1
1	2	1	1	1
2	3	1	2	2
3	2	3	3	4

* O processador 0 representa a armazenagem externa ao flow shop e o tempo correspondente representa o tempo necessário para transferir o produto da armazenagem para o processador 1

Para seqüenciar as tarefas foi utilizada uma abordagem BAB usando a heurística SMBB apresentada na Tabela 3.5 . Note-se que nesta heurística, a parcela referente ao custo futuro não incorpora os tempos de transferência

entre os equipamentos adjacentes, e nesta exemplo os tempos de transferência são importantes.

A lista de nós sondados é apresentada na Tabela 3.9. O número total de nós da árvore é 64. Neste caso foi necessário sondar 56 nós em busca da solução ótima.

Tabela 3.9 - Seqüência de execução do BAB

Sequencia Parcial	Lower Bound	Sequencia Parcial	Lower Bound
1) A	43	29) BCD	54
2) B	40	30) DCAB	57
3) C	41	31) BDA	54
4) D	39	32) BDC	54
5) DA	51	33) ABC	57
6) DB	43	34) ABD	58
7) DC	44	35) ACB	53
8) BA	49	36) ACD	55
9) BC	48	37) ADB	54
10) BD	49	38) ADC	55
11) CA	51	39) CBAD	59
12) CB	45	40) CBDA	58
13) CD	47	41) CAB	55
14) DBA	48	42) CAD	57
15) DBC	48	43) DAB	53
16) AB	49	44) DAC	54
17) AC	49	45) CDAB	59
18) AD	50	46) BCAD	61
19) DCA	49	47) ACBD	63
20) DCB	48	48) DABC	61
21) CBA	50	49) BCAD	61
22) CBD	51	50) BDAC	62
23) CDA	52	51) BDCA	61
24) CDB	51	52) ADBC	63
25) DBAC	57	53) DACB	61
26) DBCA	56	54) ADCB	62
27) DCBA	57	55) CABD	65
28) BCA	53	56) ACDB	62

A não incorporação dos tempos de transferência na parcela de estimação, faz com que o valor do limitante inferior não reflita de maneira adequada o custo associado às tarefas ainda não seqüenciadas, de forma que um maior número de caminhos pareçam igualmente promissores na construção da seqüência de tarefas. Para melhorar esta estimativa os tempos de transferência foram incorporados aos tempos de processamento de tal forma que a heurística SMBB foi utilizada na forma modificada apresentada na equação (3.9):

$$LB=C(K,M)+\sum_{i\in U}(t_{iM}+t^{*}_{iM}) \qquad (3.9)$$

com :

C(K,M)=instante de término da última tarefa da seqüência no processador M;

t^{*}_{iM}= tempo de transferência do produto i do processador M para fora do flow shop.

t_{IM}= tempo de processamento da tarefa i no processador M

U = conjunto de tarefas não seqüenciadas

K = conjunto ordenado de tarefas seqüenciadas

O número de nós sondados neste caso é menor e está apresentado na Tabela 3.10.

Tabela 3.10 - Seqüência de execução do BAB

Sequencia Parcial	Lower Bound
1) A	53
2) B	49
3) C	50
4) D	47
5) DA	50
6) DB	48
7) DC	49
8) DBA	51
9) DBC	50
10) DCA	49
11) DCB	49
12) DCAB	57
13) DCBA	56
14) BA	56
15) BC	54
16) BD	54
17) DBCA	56
18) DAB	56
19) DAC	57
20) CA	58
21) CB	51
22) CD	52
23) DBAC	57
24) CBA	51
25) CBD	53
26) CBAD	59
27) CDA	54
28) CDB	57
29) CBDA	59
30) AB	55
31) AC	57
32) AD	56
33) CDAB	59
34) BCA	57
35) BCD	56
36) BDA	57
37) BDC	56
38) CDBA	58

Para o caso de alguns processos semi-contínuos, os tempos de transferência são um detalhe fundamental para a construção do schedule (Knopf(1982), Reklaitis(1985)(1),(2)) que exigem heurísticas de agregação diferentes, pois a transferência de produtos não se dá normalmente entre processadores adjacentes mas para e da armazenagem intermediária de forma que estes tanques atuam normalmente como elementos reguladores das taxas de processamento de sorte que, dependendo do produto, podem ser criados

gargalos de processamento.

Para o caso do flow shop NIS Suhami e Mah (1981) desenvolveram expressões para o cálculo do limitante inferior em uma abordagem do tipo BAB.

3.4.3.4. Técnicas de Programação Matemática

A utilização da técnicas de Programação Matemática pressupõe que o problema tenha uma formulação matemática adequada. Para os problemas de seqüenciamento de tarefas em flow shops são conhecidas as formulações para o flow shop NW (Reddy e Rammamoorthy (1973), Pekny e Miller (1991), Birewar e Grossmann (1989)) e UIS (Baker (1973), Karimi (1988)). A formulação matemática do problema de seqüenciamento em um flow shop UIS é apresentado a seguir.

- Formulação do seqüenciamento de tarefas em um flow shop UIS como um problema de Programação Inteira.

O objetivo é seqüenciar um conjunto de tarefas em um flow shop com armazenagem intermediária infinita de forma que o tempo total de processamento (makespan) seja mínimo. Este problema pode ser formulado como um Problema de Programação Inteira tal como apresentado a seguir (Baker (1972)):

$$\min \sum_{i=1}^{N} X_{iM} \qquad (3.10)$$

sujeito a:

$$X_{j+1,k} + \sum_{i=1}^{N} t_{ik} Z_{i,j+1} + Y_{j+1,k} = Y_{j,k} + \sum_{i=1}^{N} t_{i,k+1} Z_{ij} + X_{j+1,k+1} \qquad (3.11)$$

$$\sum_{i=1}^{N} Z_{ij} = 1 \qquad (3.12)$$

$$\sum_{j=1}^{N} Z_{ij} = 1 \qquad (3.13)$$

$$Y_{jk} \text{ e } X_{jk} \geq 0 \qquad \text{e} \qquad Z_{ij} \in \{0,1\}$$

sendo:

X_{ik} = período de tempo ocioso no processador k imediatamente antes de iniciar o processamento da tarefa com posição i na seqüência, para k=2,3,....,m

Y_{ik} = período de tempo ocioso da tarefa com posição i na seqüência entre o instante em que é terminado o seu processamento no processador k e seu início no processador (k+1), para k=1,2,..,m-1

X_{i1} = 0, o primeiro processador nunca está ocioso.

Y_{iM} = 0, após o último processador a tarefa nunca está "suspensa", isto é, após a M-ésima operação de qualquer tarefa, ela deixa o flow shop.

$\sum_{i=1}^{N} t_{ik} Z_{ik}$ = tempo de processamento da tarefa com posição i na seqüência no processador k.

A equação (3.12) garante que uma tarefa ocupa uma única posição na seqüência, e a equação (3.13) garante que cada posição na seqüência é ocupada por apenas uma tarefa. A variável Z_{ij} é definida da seguinte maneira:

$$Z_{ij} = \begin{cases} 1, \text{se a tarefa i é alocada à posição j na seqüência} \\ 0, \text{caso contrário} \end{cases}$$

O makespan é igual ao tempo total ocioso no processador M mais o tempo total de processamento neste último processador, já que Y_{im} é nulo. Como o tempo total de processamento é constante, o problema de minimização do makespan pode ser escrito como minimização dos períodos ociosos,tal como apresentado na equação (3.10)(Rinnoy Kan (1976)).

A equação (3.11) representa a restrição de factibilidade do flow shop apresentada na Figura 3.5.

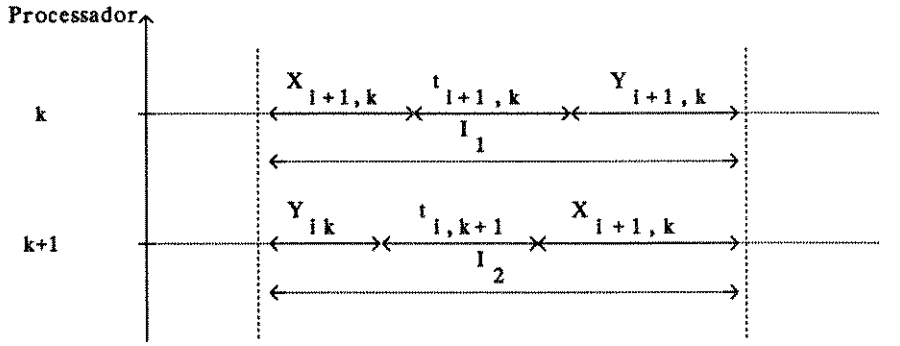


Figura 3.5 - Restrição de factibilidade do flow shop UIS ($I_1=I_2$) (Equação 3.11)

3.4.3.5. Regras Heurísticas para seqüenciamento de tarefas

As heurísticas são algoritmos ou regras desenvolvidos com o objetivo de fornecer rapidamente uma solução do problema, ou pelo menos em um intervalo de tempo bem inferior ao que seria gasto usando as técnicas apresentadas nas secções (3.4.3.3) e (3.4.3.4).

Para o caso do flow shop existem poucos algoritmos ou regras heurísticas(Dannenbring (1978), Ashour(1970)) normalmente utilizados, e foram desenvolvidas com o objetivo de gerar seqüências de permutação.

Na Tabela 3.11 são apresentadas as principais heurísticas usadas para o seqüenciamento em flow shops. Elas exigem que os tempos de transferência de produto entre equipamentos adjacentes e os tempos de preparação sejam nulos, ou independentes da seqüência, e incorporados aos tempos de processamento. Todas estas heurísticas foram desenvolvidas com o objetivo de minimizar o makespan.Outras regras heurísticas para o seqüenciamento de tarefas em flow shops podem ser obtidas em Booth e Turner (1987).

É interessante notar que das quatro heurísticas apresentadas na Tabela 3.11, três são baseadas em princípios semelhantes. Estas heurísticas recorrem ao artifício de reduzir o problema de seqüenciamento em um flow shop com M processadores a um problema fictício de apenas dois processadores. Nas heurísticas RA e CDS este problema fictício é resolvido através do recurso ao Algoritmo de Johnson (Seção 3.4.3.2).

A análise das referências originais em que estas regras heurísticas foram apresentadas revela que não há qualquer justificativa teórica ou de ordem prática que justifique a qualidade das abordagens propostas.

A heurística NEH (Nawaz et al. (1983)) é uma das que oferece melhor desempenho no seqüenciamento de tarefas em um flow shop UIS e seu algoritmo resumido e um exemplo de aplicação são apresentados a seguir.

Tabela 3.11 - Heurísticas para o seqüenciamento de tarefas em flow shops (Dannenbring (1979), Campbell et al (1970), Ashour (1970), Nawaz et al. (1983) Booth e Turner (1987))

1) Heurística de Dannenbring (RA - Rapid Acess)

$$t^{*}_{i1} = \sum_{j=1}^{M} (M-j+1)\, t_{ij} \qquad t_{i2} = \sum_{j=1}^{M} j\, t_{ij}$$

O subproblema de dois estágios é resolvido usando o Algoritmo de Johnson

2) Heurística de Petrov

$$t_{i1} = \sum_{j=1}^{k_1} t_{ij} \qquad t_{i2} = \sum_{j=k_2}^{M} t_{ij}$$

com $k_1 = k_2 = \frac{M}{2} + 1$ se M é ímpar

com $k_1 = \frac{M}{2}$, $k_2 = \frac{M}{2} + 1$ se M é par

Formar sequencias com $(t_{i2} - t_{i1})$ em ordem crescente

3) Heurística de NEH (Nawaz, Enscore e Ham)

Apresentada a seguir

4) Heurística de Campbell-Dudek-Smith (CDS)

Gerar (M-1) problemas artificiais de dois estágios utilizando as seguintes expressões para o cálculo dos tempos de processamento em cada estágio:

$$t^{k}_{i1} = \sum_{j=1}^{k} t_{ij} \qquad t^{k}_{i2} = \sum_{j=M-k+1}^{M} t_{ij}$$

Os subproblemas de dois estágios são resolvidos usando o Algoritmo de Johnson escolhendo a seqüência que fornecer o menor valor do makespan

Heurística de NEH (Nawaz et al. (1983))

Passo 1– Ordenar as tarefas de acordo com os valores decrescentes de :

$$T_i = \sum_{j=1}^{M} t_{ij}$$

gerando a seqüência $i_1, i_2, i_3, \ldots, i_k, \ldots, i_N$

Passo 2 - Entre as seqüências parciais geradas pelos dois primeiros produto $:i_1-i_2 e i_2-i_1$, selecione a melhor (menor makespan), que servirá então de ponto de partida para a construção de seqüências derivadas.

Passo 3 - Dada uma seqüência parcial com k produtos, $i_1, i_2, \ldots i_k$ calcule o makespan para cada uma das seqüências parciais:

$i_{k+1}\ i_1 i_2 i_3 \ldots i_k$

$i_1 i_{k+1} i_2 i_3 i_k$

.

$i_1 i_2 i_3 \ldots i_k i_{k+1}$

Passo 4 - Selecione a seqüência com menor makespan

Passo 5 - Repita os passos 3 e 4 até que todas as tarefas estejam seqüenciadas.

Tabela3.12 - Dados para o exemplo - Heurística NEH

Produto	Unidade					T_i
	1	2	3	4	5	
A	5	9	8	10	1	33
B	9	3	10	1	8	31
C	9	4	5	8	6	32
D	4	8	8	7	2	29

Exemplo de aplicação da heurística de NEH

- Ordenamento inicial : A - C - B - D
- Construção da seqüência:

Seqüência	Makespan
A - C	46
C - A	42→ Selecionada
C - A - B	50 → Selecionada
C - B - A	51
B - C - A	51
C - A - B - D	58
C - A - D - B	58
C - D - A - B	57
D - C - A - B	54 → Seqüência escolhida

Estudos da qualidade das soluções geradas através do uso de heurísticas são apresentados em Booth e Turner (1987) e Woollam (1986)

(Panwalker e Iskander (1977)). Em função da degeneração da qualidade destas soluções à medida que a dimensão do problema cresce, estas regras não são utilizadas para a geração de solucoes finais, mas sim como geradoras de soluções iniciais para serem aprimoradas a posteriori através do uso de algoritmos de refinamento destas soluções. Um exemplo de algoritmo de refinamento é a estratégia RAES - Rapid Acess Extended Search (Dannenbring (1978)), utilizado para refinar a solução gerada pela heurística de Dannenbring (RA). Os passos principais da estratégia RAES são apresentados a seguir.

Heuristica RAES

Passo 1 - Gerar a seqüência inicial pela heurística RA

Dado :
1 - 2 - 3 - .....- (k-1) - k - ....- (N-1) - N

Passo 2- Propor intercambios entre tarefas adjacentes e avaliar o makespan

2 - 1 - 3 - .....- (k-1) - k - ....- (N-1) - N
1 - 3 - 2 - .....- (k-1) - k - ....- (N-1) - N
.
.
.
1 - 2 - 3 - .... - k - (k-1) - ....- (N-1) - N
1 - 2 - 3 - .... - (k-1) - k - ....- (N-1) - N

Passo 3 - Selecionar a melhor e continuar

A estratégia do tipo RAES requer:
1) O desenvolvimento de um algoritmo eficiente de controle de seqüências já sondadas;
2) O cálculo do makespan de todas as seqüências candidatas.

Estas duas características aproximam a estratégia do tipo RAES dos métodos de busca sem que haja qualquer garantia de ganho em termos de tempo necessário para a busca de soluções com relação a estes últimos.

3.4.3.6. Abordagens para o tratamento de restrições estáticas

As principais restrições estáticas são: restrições tecnológicas de precedência e restrições no prazo de entrega.

a) Restrições de prazo de entrega

O problema do flow shop com penalidade no prazo de entrega foi estudado por diversos autores (Musier e Evans (1987), Karimi et al (1991), (McMahon e Florian (1975), (Posner (1988)). A maior parte dos trabalhos desenvolvidos utilizam a abordagem BAB com modificações do limitante inferior proposto originalmente por Shwrimer (1972) para o flow shop UIS com dois processadores.

A limitação dos diversos algoritmos ao problema do flow shop com dois processadores evidencia a complexidade do problema de restrições no prazo de entrega.

Algoritmo de Shwrimer (1972)

O objetivo é minimizar o custo total dos "atrasos". Este custo está

associado à defasagem "positiva" ($c_{i2}-d_i$), sendo c_{i2} o instante de término da tarefa i no processador 2 e d_i o prazo de entrega da tarefa i. A penalidade, p_i, é definida como $/unidade de tempo.

As hipóteses admitidas são:
- São considerados apenas os schedules de permutação;
- Não é permitida a preempção de tarefas;
- Os tempos de processamento são determinísticos;
- Os set-up e tempos de transferencia são desprezíveis;
- Não há restrições de precedência.

A função objetivo será:

$$P = \sum p_i \max[0, c_{i2}-d_i] \qquad (3.14)$$

A estratégia BAB consiste em calcular em cada nó da árvore um custo total (P) composto da soma de duas parcelas:

$$P=P[S]+P[S']$$

onde:
P[S]= penalidade (custo) associada ao conjunto de tarefas seqüenciadas (S)
P[S']= estimativa da penalidade associada ao conjunto de tarefas não seqüenciadas (S')

O cálculo de P[S'] é feito baseado na estimativa dos instantes de término (C_{ij}) das tarefas ainda não seqüenciadas.

Por se tratar de um sistema de apenas dois processadores, cada um deles pode atuar como gargalo do processo.

A estratégia consiste então em estimar o instante de término de cada tarefa não seqüênciada considerando que cada processador pode atuar como gargalo do processo e portanto atrasar o término de cada uma das tarefas.

As relações necessárias para realizar estas estimativas são: relações de recorrência para o cálculo dos "completion time" em cada processador, e relações para a estimativa do makespan.

O tempo de término das tarefas é dado por:

$$C_{i1}=C_{(i-1)1}+t_{k_i 1} \qquad (3.15)$$

$$C_{i2}=\max[C_{i1},C_{(i-1)2}]+t_{k_i 2} \qquad (3.16)$$

Se a unidade 1 é o gargalo o limitante inferior (LB_1^*)será:

$$C_{i1}=C_{(i-1)1}+t_{k_i 1} \qquad (3.17)$$

$$C_{i2}=C_{i1}+t_{k_i 2} \qquad \Rightarrow LB^*_{1,i} \qquad (3.18)$$

Se a unidade 2 é o gargalo, o limitante inferior (LB_2^*) será:

$$C_{i1}=C_{(i-1)1}+t_{k_i 1} \tag{3.19}$$

$$C_{i2}=C_{(i-1)1}+t_{k_i 2} \quad \Rightarrow LB^{*}_{2,i} \tag{3.20}$$

O limitante inferior será então:

$$LB=P[S]+\sum p_i \quad \max_{i \in K} [LB^{*}_{1,i}, LB^{*}_{2,i}] \tag{3.21}$$

Uskup e Smith (1975) propuseram igualmente um procedimento BAB associado a testes de factibilidade para o seqüenciamento de tarefas em flow shops UIS com dois estágios e restrições no prazo de entrega e set-up dependente da seqüência.

A vantagem de utilização de métodos de busca para tratar prazos de entrega se prende ao fato que eles permitem relaxamento da restrição através da cobrança de penalidades, o que não é possível quando se recorre à formulação matemática.

b) Restrições de Precedência

Este tipo de restrição pode ser resolvida pelo recurso à formulação matemática e métodos de busca. A adição de restrições tecnológicas de precedência às técnicas de enumeração ou de programação matemática pode ser complexa e o problema está discutido em Lenstra e Rinnooy Kan (1978) e Hoitmot et al (1990).

No caso de utilizar métodos de busca, as restrições de precedência surgem na forma de nós infactíveis que resultam no abandono destes ramos. Na programação matemática surgem na forma de desigualdades que limitam os "completion times".

3.5. Abordagens para tratamento de restrições dinâmicas

As abordagens apresentadas para o problema de seqüenciamento de tarefas mostram que o problema de alocação se restringe a definir os instantes de término de cada tarefa a fim de estabelecer o perfil de ocupação dos processadores.Nestes casos o cálculo dos instantes de término das tarefas (completion time) é dependente apenas da política de processamento no shop, tais como UIS,NIS e NW(Anexo A).

Nos problemas de scheduling, isto é, naqueles problemas em que haja limitação na oferta de armazenagem intermediária e recursos comuns, a definição dos instantes de início da ocupação dos processadores e de sua liberação é feita com base na satisfação de restrições. Conseqüentemente seqüenciamento de tarefas e perfil de alocação das tarefas aos processadores são atividades indissociáveis a menos que sejam aceitos relaxamentos das restrições. O cálculo do completion time é importante na medida que:

1) As funções objetivo normalmente empregadas nos problemas de scheduling são funções do tempo. A função objetivo mais utilizada é o makespan que é o tempo total de ocupação do shop. A utilização de outras

funções de custo, tais como custo de utilização da armazenagem, número de tarefas atrasadas etc, exigem igualmente o cálculo dos "completion time".

2) A definição do "tempo de residência" de cada uma das operações de cada tarefa em cada processador dependerá da capacidade de acomodação das demandas temporais globais dos recursos compartilhados. O Anexo A mostra a relação existente entre cálculo de "completion time" e a política de alocação de recursos.

Em resumo, o atual estado da arte do problema do scheduling em flow shops revela que:

1) Existem várias propostas de abordagens que em princípio, tem o objetivo de resolver o problema de alocação de recursos. No entanto estas propostas não foram desenvolvidas o suficiente a ponto de solucionar o problema em questão (Steffen e Greene (1986), Reklaitis et al. (1982)).

2) A armazenagem intermediária é o recurso compartilhado que tem merecido maior atenção na literatura específica de Engenharia Química. Mas mesmo neste caso as propostas de solução são ainda incipientes.

3) Em áreas correlatas, tais como scheduling em job shops e sistemas distribuídos, tem surgido abordagens alternativas cujo potencial de utilização para o scheduling em flow shops merece ser pesquisado.

Embora não haja propostas para a solução do problema de seqüenciamento de tarefas em flow shops na presença de restrições na oferta de recursos compartilhados, existem caminhos para a solução do problema que podem ser explorados. Algumas das alternativas de solução do problema de scheduling que serão discutidas a seguir, e são:

1) Discussão da extensão das técnicas clássicas de seqüenciamento para a solução do problema de scheduling.

2) Apresentação e discussão de trabalhos voltados para o scheduling em job shops que utilizam conceitos de planejamento hierárquico e conceitos de janelas de demanda.

3) Apresentação e discussão de técnicas de scheduling em job shops orientado por recursos e orientado por tarefas.

3.5.1. Extensões das técnicas sem restrições

Na Seção 3.4.3 foram discutidas as principais abordagens para a solução dos problemas de seqüenciamento que são: Programação Matemática, Métodos de Busca e Métodos Heurísticos.

As dificuldades associadas à utilização de cada uma destas abordagens para a solução dos problemas de scheduling e as eventuais limitações que possam existir, serão discutidas a seguir. É importante esclarecer que a discussão destas técnicas será baseada quase totalmente em hipóteses viáveis para a solução do problema de scheduling, já que não há praticamente discussões e trabalhos publicados na literatura sobre este assunto.

3.5.1.1. Formulação como Problema de Programação Linear Mista

A opção pela utilização de Técnicas de Programação Matemática para a solução de problemas de scheduling deve ser feita após uma análise cuidadosa do problema contemplando os seguintes pontos:

- As restrições que afetam a oferta e a demanda de recursos comuns devem ser expressas através de relações matemáticas;

- Qual o número de variáveis necessárias para caracterizar o problema, especialmente o número de variáveis sujeitas às restrições de integralidade.

Para fins de discussão deste tipo de abordagem, os recursos serão tratados como recursos compartilhados e armazenagem intermediária separadamente.

Quando existe compartilhamento de recursos, a restrição que se põe imediatamente é que a demanda temporal de recursos deve ser no máximo igual à oferta destes mesmos recursos. A relação (3.1) representa este tipo de restrição. No entanto, na forma como ela está apresentada pode-se visualizar muito pouco das dificuldades em utilizá-la em problemas de scheduling usando a abordagem de Programação Matemática. Esta relação concentra um grande conjunto de informações. De fato ela significa que:

a) A demanda acumulada de recurso compartilhado em cada instante deve ser no máximo igual à oferta de recursos e em cada instante a demanda não deve ultrapassar a oferta de recursos;

b) A alocação de uma tarefa a um processador exige que o volume de recursos disponível em um intervalo de tempo igual ao tempo de processamento da tarefa no processador deve ser no mínimo igual ao volume de recursos exigido para a execução desta tarefa.

A utilização da Programação Matemática exige então que as condições a) e b) sejam apresentadas na forma de relações matemáticas.

A condição a) pode ser expressa através da relação:

$$U^l = \sum_{i \in K} \sum_{j=1}^{M} \int_{t=0}^{t=C(K,M)} Q_{ij}^l(t).Z_{ij}(t)dt \qquad (3.22)$$

onde:

U^l = Demanda global do recurso l no intervalo $0 \le t \le C(K,M)$

Q_{ij}^l = Demanda intantânea de recurso l para a execução da tarefa i no processador j

$$Z_{ij} = \begin{cases} 1, \text{se a tarefa i está programada no processador j no instante t} \\ 0, \text{caso contrário} \end{cases}$$

A condição b) exige que sejam desenvolvidas expressões matemáticas em função dos "completion time". A dificuldade associada ao desenvolvimento de expressões deste tipo estão discutidas no Anexo A, e justificam o fato de não existir até o presente uma formulação matemática para o problema.

O caso da armazenagem intermediária inter-estágios foi estudado por Ku e Karimi (1988), que desenvolveram a modelagem matemática como um problema MILP para a minimização do makespan, a qual é apresentada a seguir:

Função Objetivo

$$\min_{N} C_{NM}$$

$$C_{ij} - C_{(i-1,j)} - \sum_{k=1} t_{kj} X_{ki} \ge 0 \quad 1 \le j \le M \ , \ 1 \le i \le N \qquad (3.23)$$

$$C_{ij} - C_{(i,j-1)} - \sum_{k=1}^{N} t_{kj} X_{ki} \geq 0 \qquad 2 \leq j \leq M \ , \ 1 \leq i \leq N \qquad (3.24)$$

$$C_{ij} - C_{(i-z_j-1,\, j+1)} \geq 0 \qquad 1 \leq j \leq M-1 \ , \ 1 \leq i \leq N \qquad (3.25)$$

$$\sum_{j=1}^{N} X_{ij} = 1 \qquad 1 \leq i \leq N \qquad (3.26)$$

$$\sum_{i=1}^{N} X_{ij} = 1 \qquad 1 \leq j \leq N \qquad (3.27)$$

onde:

z_j = número de tanques de armazenagem disponíveis após o estágio j

$$X_{ij} = \begin{cases} 1 \ , \text{ se o produto i ocupa a posição j na seqüência} \\ 0 \ , \text{ caso contrário} \end{cases}$$

Os autores alegam que a utilização de uma formulação deste tipo é mais conveniente em relação à abordagem BAB na medida que existem pacotes para resolver problemas MILP, enquanto que o algoritmo BAB deve ser totalmente codificado.

Por outro lado os autores afirmam que a abordagem proposta consome um tempo de computação bastante elevado, sendo seu uso recomendado para problemas da ordem de até MxN=30. No entanto é bastante comum na indústria química ordens superiores à citada. Em função disto os autores propuseram simplificações na formulação do problema, relaxando as restrições de integralidade, e arredondando os valores de X_{ij} não zeros. Esta estratégia apresentou uma grande degeneração da solução com relação à solução obtida na formulação original. Como alternativa, os autores utilizaram esta abordagem simplificada para gerar uma solução inicial a ser posteriormente sondada através de estratégias do tipo RAES.

Para o caso da armazenagem compartilhada entre estágios a formulação matemática do problema se aproxima bastante do problema dos recursos comuns já discutido anteriormente. Neste caso a restrição de factibilidade da utilização da armazenagem disponível depende de uma análise global da demanda de armazenagem (Anexo A), e não apenas de uma análise da demanda entre dois estágios consecutivos. Por esta razão não existe até o presente a formulação de uma abordagem que utiliza ferramentas de programação matemática.

3.5.1.2. O Branch and Bound cl#ssico e o incremento em backtracking

A utilização de uma abordagem BAB para o scheduling em flow shops pode ocorrer em dois cenários diferentes que serão denominados:

-Cenário de "scheduling";

-Cenário de "seqüenciamento".

A)O cenário de "scheduling" é definido como aquele em que o objetivo é sequenciar as tarefas obedecendo as restrições de limitação na oferta de recursos comuns e, simultâneamente, otimizando um critério de desempenho do sistema.

A grande dificuldade e principal limitação associada à utilização desta abordagem reside na eficiência da atividade de controle do método BAB a qual exige o cálculo da função custo. A eficiência desta atividade

está fortemente relacionada com a qualidade da estimação do custo associado a cada nó da árvore. As parcelas que compõem a função custo são normalmente calculadas da seguinte forma:

- A primeira parcela reflete o custo da seqüência parcial no nó, já incorporando o efeito sobre a função custo resultante das decisões de alocação dos recursos (utilidades, armazenagem intermediária etc);
- A segunda parcela reflete uma estimativa do custo mínimo ainda necessário para sequenciar o restante das tarefas. É desejável que esta estimativa seja feita considerando que a oferta de recursos é limitada.

Uma pesquisa da literatura mostra que não há limitantes com as características apontadas acima. Neste caso, pode-se empregar um limitante capaz de estimar o makespan considerando que a oferta de recursos é ilimitada. Na prática este procedimento é equivalente a calcular a segunda parcela do limitante usando qualquer uma das heurísticas apontadas na Tabela 3.5.

No Capítulo 4 é apresentada uma proposta para cálculo desta segunda parcela em que as restrições são satisfeitas e o critério de desempenho é otimizado.

B) O cenário de "seqüenciamento" é composto de duas fases. Na primeira fase é gerada uma seqüência de tarefas através do emprego de uma abordagem do tipo BAB, otimizando o critério de desempenho do sistema escolhido sendo que as restrições na oferta dos recursos comuns e armazenagem intermediárias são ignoradas. Na segunda fase é calculado o perfil de demanda dos recursos comuns e armazenagem intermediária. Os conflitos detectados na construção destes perfis devem então ser resolvidos nesta segunda fase.

A factibilização a posteriori implica em tomar decisões de alocação, isto é, implica em decidir, na ocorrência de um conflito, qual operação será "deslocada" ou seja, qual evento deverá ser atrasado. Estes atrasos podem apresentar reflexos temporais em eventos posteriores e, consequentemente no valor do makespan da seqüência. Estas decisões de alocação são tomadas com base em estratégias previamente estabelecidas tais como a estratégias de prioridade do evento e prioridade do produto (Anexo B). Neste procedimento não há qualquer compromisso com a otimalidade da solução gerada. Uma alternativa pode ser a melhoria eventual da solução obtida através do emprego de técnicas de sondagem das vizinhanças.

As seqüências geradas através deste procedimento podem então ser utilizadas como seqüências candidatas a exploração de soluções vizinhas para verificar a possibilidade de redução do valor da função objetivo. Estas sondagens são normalmente realizadas através do intercâmbio de tarefas adjacentes. É claro que esta sondagem não garante que será encontrada uma solução ótima, nem mesmo que será melhorada a solução atual, pois pode ser necessário um número elevado de intercâmbios para ser obtida alguma melhoria da função objetivo. Deve-se ainda ter em conta que, normalmente, existe mais de um recurso compartilhado. Pode ocorrer que, na realização destes intercâmbios, alguns conflitos sejam resolvidos mas que também sejam criados novos conflitos, o que acaba por demandar várias operações de compatibilização dos perfis de oferta e demanda dos diferentes recursos. Por esta razão normalmente não é realizada a sondagem das vizinhanças, sendo aceita a solução original.

A diferença na qualidade das soluções de cada um destes cenários dependerá da importância e frequência dos conflitos que acontecerem durante o desenvolvimento do scheduling

Naqueles casos em que a demanda de recursos não for crítica, isto é, as penalidades associadas à violação das restrições forem pequenas e/ou os períodos de violação da oferta sejam reduzidos, então seus reflexos no valor do makespan devem ser igualmente pequenos. Neste caso a diferença na qualidade das soluções geradas nos dois cenários não deve ser significativa.

No entanto, se a violação das restrições impõe penalidades severas na forma de crescimento do critério de desempenho adotado e/ou a compatibilização dos períodos de violação das restrições exigirem a introdução de atrasos consideráveis no início de excução das operações refletindo em um aumento elevado do makespan, então neste caso o cenário de seqüenciamento deve ser descartado.

Para distinguir se o problema real se enquadra em uma das categorias acima é desejável dispor de um sistema capaz de realizar análises de capacidade para auxiliar a decidir em qual das categorias acima o problema se enquadra. Deve-se notar que um sistema de geração de schedules factíveis deve em princípio conter abordagens para as duas categorias de problemas, pois uma das principais características dos sistemas flexíveis é a variabilidade dos perfis de demanda dos produtos e portanto da demanda de recursos.

No caso em que seja indicado o uso de um cenário de scheduling, se o limitante inferior for calculado com base em uma heurística inadequada, a conseqüência imediata será o aumento da freqüência de backtracking à medida que aumente a limitação na oferta de recursos.

A ilustração da utilização destes dois cenários está apresentada na Seção 4.2.

3.5.1.3. Inclusão nas Técnicas Heurísticas de Heurísticas de Alocação

Até o presente não foram desenvolvidas heurísticas para o caso do flow shop com recursos comuns.

A estratégia empregada pode ser a utilização de heurísticas desenvolvidas para o flow shop UIS juntamente com as expressões de "completion time" apresentadas no Anexo A e em seguida sondar soluções vizinhas que possam eventualmente melhorar o valor do makespan de forma semelhante ao cenário de seqüenciamneto. Um exemplo deste tipo é o algoritmo IMS apresentado a seguir.

Analise do tempo ocioso (Algoritmo IMS) (Karimi, 1988)

Este algoritmo consiste em construir a matriz E dos tempos ociosos e analisar quais "ligações" de pares de tarefas i-j são candidatas a serem alteradas através da análise do termo e_{i-j}, ou seja, o tempo ocioso total nas unidades quando o produto j segue o produto i.

Cada par i-j é uma ligação, e cada uma delas contribui com um período ocioso. A idéia é reduzir o makespan reduzindo o período ocioso, através de rupturas sucessivas das piores ligações.

O procedimento é:
1- Gerar uma seqüência inicial;
2- Das (n-1)ligações i-j desta seqüência, escolher aquela que apresenta o maior tempo ocioso;
3- Aplicar um dos algoritmos de modificações de seqüência, por exemplo uma estratégia do tipo RAES, e escolher aquela com menor makespan para ser analisada em termos de tempos ociosos. Retornar ao passo 2.

Este procedimento se baseia em, a partir de uma seqüência inicial,gerar a Carta de Gantt e sondar as vizinhanças desta solução para verificar se há possibilidade de melhorar o valor da função objetivo. No procedimento equivalente usado no seqüenciamento de tarefas apresentado na Seção 3.5.1.3. já foram apresentados os principais inconvenientes na utilização desta abordagem.

A utilização de técnicas heurísticas, bem como a utilização de uma abordagem BAB de duas fases, implica que a posteriori da atividade de seqüenciamento sejam compatibilizados os perfis de oferta e demanda deste recurso de forma a solucionar o conflito. Esta forma de solucionar o problema implica em que sejam utilizadas estrategias de alocacao para resolver o conflito quando este ocorrer. Algumas destas estratégias e de procedimentos de alocação estão apresentadas no Anexo B.

3.5.2. Utilização do conceito de janela de demanda

Nas áreas correlatas de scheduling em job shops tem havido propostas de abordagens para a solução de problemas com limitação na oferta de recursos compartilhados pois, no caso do job shop, os processadores são os recursos compartilhados por excelência.Dois trabalhos significativos destas áreas foram escolhidos para fornecer um panorama das ferramentas e metodologias básicas que são utilizadas pois, sendo um problema semelhante ao do scheduling em flow shops, podem ser extraídas algumas contribuições importantes, principalmente com relação aos mecanismos de busca e controle empregados.

Egli e Rippin (1986) e Fox (1983) abordam o problema do scheduling em job shops com a preocupação de desenvolver sistemas capazes de, hierarquicamente, conduzir à construção de soluções implementáveis usando métodos de busca. Em ambos os trabalhos a atividade de scheduling propriamente dita é precedida de uma atividade de "pré-busca" na qual são consideradas as restrições de natureza tecnológica e outras restrições especificadas pelo usuário para efeito de eliminação daquelas soluções que não poderão ser aceitas sob quaisquer condições (normalmente restrições tecnológicas e restrições de precedência). O resultado imediato desta atividade é a redução do espaço de soluções factíveis.

Mesmo após a redução do espaço de soluções factíveis restará normalmente um número elevado de "ordenamento de operações" factíveis. A busca da solução é orientada através de um ordenamento parcial das operações decorrente da utilização do conceito de janela de demanda.

A janela de demanda é uma janela de tempo factível tecnológicamente 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 reflexo do 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.

Na Figura 3.5 estão apresentadas as janelas de demanda de uma tarefa formada por três operações com duração de três unidades de tempo cada uma, e prazo de entrega igual a 15.

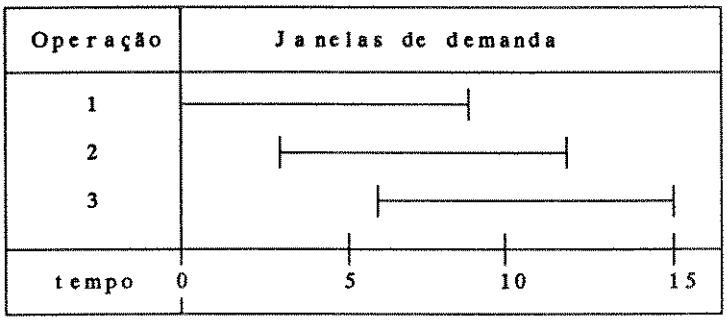


Figura 3.5 - Exemplo de janela de demanda

3.5.2.1. Técnica de Egli e Rippin

Egli e Rippin propõem um sistema formado por duas fases:

1) Na primeira fase é definido um ordenamento parcial das operações através da utilização de uma estratégia de eliminação de ordenamentos parciais que conduzirão a soluções infactíveis. Estes ordenamentos parciais obedecem as restrições tecnológicas de precedência e os prazos de entrega.Posteriormente são criadas as janelas de demanda de cada operação. Como resultado desta primeira fase são criados pré-schedules em que cada uma das operações é alocada no final de sua respectiva janela de demanda, os quais serão detalhados em um nível posterior de alocação. Quando as tarefas são alocadas o mais tarde possível, os custos de armazenagem intermediária são minimizados.

2) Na segunda fase são identificados os eventuais conflitos decorrentes da estratégia de alocação adotada na primeira fase. O gerenciamento dos conflitos é feito através da administração de deslocamentos (shifts) temporais factíveis ou aceitáveis das operações geradoras dos conflitos, desde que contribuam para a otimização dos critérios de otimização adotados. A freqüência de backtracking é reduzida pela aceitação de schedules parciais que são factibilizados através da incorporação de penalidades na função objetivo.

3.5.2.2. Técnica de Fox (ISIS)

O sistema de alocação proposto por Fox utiliza conceitos bastante semelhantes aos empregados por Egli e Rippin. Fox utiliza os mesmos conceitos de janelas de demanda, cobrança de penalidades associadas ao relaxamento de restrições e recorre a um esquema de pré-busca para eliminação de soluções infactíveis a priori pois violam as restrições de precedência e prazos de entrega. A principal diferença entre os dois trabalhos reside na estratégia de alocação.

No caso do sistema ISIS a escolha da próxima operação a ocupar o

shop é feita pelo usuário, ou com base em parâmetros definidos pelo usuário. Esta estratégia é equivalente a adotar prioridades entre as diferentes tarefas a serem processadas. Em seguida são criadas as janelas de cada operação para proceder à alocação.

A decisão do instante em que a operação deve ser alocada é tomada com base em dois tipos de operadores de reserva: i) "eager reserver", que aloca a operação o mais cedo possível procurando minimizar o tempo de residência (makespan); ii) "wait and see reserver", que aloca a operação procurando minimizar o tempo de ocupação da armazenagem intermediária. Para cada operação existem então duas soluções parciais baseadas em cada um dos operadores. Cada solução parcial é sondada recorrendo a um método de busca do tipo Beam Search.

Os nós desta árvore podem ser descartados através de dois mecanismos: ou são infactíveis por violação de alguma restrição "rígida", ou são eliminados por excederem a largura da busca especificada a priori para desenvolvimento da busca (Beam Search). ÍÀ medida que vão se sucedendo as alocações, são criadas novas restrições que devem ser propagadas, pois a alocação de uma operação pode provocar redução nas janelas temporais das operações ainda não alocadas.A atividade de busca é crucial na construção de schedules e, no caso do job shop, o número de alternativas de ramificações de cada nó é em geral bastante elevado, de forma que o autor propõe um mecanismo de escolha da próxima operação a ser alocada com base em parâmetros de prioridade pré-estabelecidos. Na atividade de controle onde se calcula o custo associado a cada nó da árvore é, aparentemente, considerado apenas o custo relativo à solução parcial do nó, sem a adição de qualquer estimativa de custo relativa às operações ainda não alocadas.

3.5.3. Partição do problema: Scheduling orientado por tarefas e Scheduling orientado por recursos

3.5.3.1. Conceito Básico

As abordagens recentes ao problema de scheduling partem da constatação de que, quando as restrições são importantes, a colocação do problema como sendo a de minimizar uma função de custo temporal sujeita a restrições pode não ser a mais adequada. Em seções anteriores discutiu-se que estas técnicas estão fortemente orientadas para a minimização da função de custo,dando ao problema de satisfação das restrições sobre recursos comuns (à exceção dos processadores), uma importância secundária.

Qualquer que seja a abordagem utilizada podem ser identificados inconvenientes:

-A utilização de métodos aproximados leva frequentemente à construção de "soluções preliminares", com custo computacional elevado, as quais devem ser profundamente modificadas para satisfazer as restrições.
- A utilização de métodos que conduzem à solução exata apresentam em geral freqüência de backtracking elevada, que se traduz igualmente em tempo computacional elevado.

O questionamento óbvio é se uma análise preliminar dos problemas com as restrições não teria permitido orientar a resolução do problema de forma a não obter "soluções preliminares" altamente infactíveis.

A idéia básica passa a ser então a de que o conflito pela utilização dos recursos críticos teria que ser resolvido, quando existir, como um problema específico, e que esta solução deveria fornecer o quadro dentro do qual se procede a minimização de uma função de custo temporal.

Esta problema tem sido abordado na literatura ou como um problema de otimização multiobjetivo ou como um problema de planejamento hierárquico.

Neste trabalho será empregada a abordagem de considerá-lo como um problema de planejamento a ser decomposto em dois subproblemas. Para esta decomposição se seguem duas perspectivas: uma orientada pelo encadeamento das operações de cada tarefa e a outra orientada pela coexistência no tempo de operações ou tarefas com demanda simultânea por recursos comuns limitados. Como é apresentado em (Fox,1990) os engargalamentos deveriam, em princípio, ser resolvidos com uma metodologia orientada pela satisfação das restrições. Através desta metodologia seriam primeiramente alocadas as operações com grande potencial de originar problemas com os recursos compartilhados, a alocação das outras operações seria feita com o recurso a uma metodologia orientada para a minimização de um critério de custo temporal onde o encadeamento da alocação das operações fosse decisivo.

As dificuldades com uma abordagem com estas características são basicamente duas:

- a determinação de quais recursos compartilhados e em quais intervalos de tempo a sua demanda se torna crítica, o que obviamente é função da alocação no tempo das operações (que é o objetivo final);
- a metodologia para as decisões de alocação após detectado um problema com recursos compartilhados, dado que esta decisão vai ter impacto sobre o tempo total necessário para o processamento de todas as tarefas.

Estes aspectos serão analisados nas seções 3.5.3.2 - 3.5.3.4.

Existem exemplos de sistemas para os quais se tem procurado desenvolver uma abordagem hierárquica de planejamento. Na literatura de aplicações de Inteligência Artificial aos problemas de Planejamento em geral denominam-se sistemas de planejamento multi-agentes. Estes sistemas tem como objetivo a síntese de um conjunto de planos de produção que atingem um conjunto de objetivos parcialmente interligados. No problema que interessa aqui os agentes são as tarefas constituídas por operações que devem ser alocadas no tempo.

Existem propostas de dois tipos de decomposição (Smith, 1985):

- A decomposição "tradicional" orientada por tarefas. Este tipo de decomposição é certamente a abordagem mais adequada para resolver os problemas originados por restrições originadas pelo encadeamento das tarefas, como por exemplo relações de precedência e, até certo ponto, restrições em armazenagem intermediária (no que se refere à duração da utilização dela). São as restrições denominadas **intra-tarefa** ou **intra-ordem** (Smith, 1985)(Keng, 1988).
-A decomposição baseada nos recursos compartilhados. Uma decomposição do segundo tipo será mais eficiente para resolver conflitos entre operações pertencentes a diferentes tarefas que demandam recursos comuns. São as restrições denominadas **inter-tarefa** ou **inter-ordem**, como por exemplo as restrições sobre utilidades, mão-de-obra e quantidade de armazenagem intermediária.

O problema real interliga estas duas decomposições: o scheduling de operações em tarefas pode gerar conflitos por recursos comuns num determinado intervalo de tempo, e a alocação de um recurso para uma operação num certo intervalo de tempo contribui para definir o scheduling final da tarefa à qual pertence esta operação. A proposta é de particionar a abordagem do problema com o objetivo de chegar a uma solução final mais rapidamente.

Um elemento essencial num sistema deste tipo é a análise do caráter crítico da demanda de recursos compartilhados e a decisão de proceder a uma alocação orientada pelos recursos num determinado intervalo de tempo.

Este procedimento deve ser necessariamente associado a procedimentos que garantam a factibilidade tecnológica das alocações, que constituem a etapa de propagação de restrições (Sacerdoti, 1974). Sem uma utilização controlada destes procedimentos poderá ocorrer a alocação indiscriminada de muitas operações, infactibilizando o schedule. Dada a dimensão do problema é bastante frequente a necessidade de estabelecer uma escala de prioridade sobre as restrições a serem satisfeitas e implementar o sistema de uma forma interativa que permita ao usuário a modificação de dados e a obtenção de soluções alternativas.

A importância relativa das restrições sobre os recursos comuns sugere uma característica adicional interessante para a metodologia em questão. Trata-se de resolver o problema de uma forma hierarquizada onde inicialmente são consideradas apenas as restrições mais importantes, sendo as demais incorporadas sucessivamente. O interessante de um sistema com esta característica é que o modelo do shop utilizado nos diferentes níveis pode variar, tendo um nível de detalhamento em cada nível compatível com as restrições alocadas nele. Um modelo mais agregado implicará num esforço computacional maior. Uma linguagem orientada a objetos parece muito adequada para a implementação de uma metodologia deste tipo na medida em que facilita a definição e utilização de objetos com diferentes graus de agregação. Este tipo de abordagem não existe ainda na literatura.

3.5.3.2. Quantificação do caráter crítico de uma operação

Em (Keng,1988) apresenta-se uma definição da "criticalidade" de uma operação entendendo como operação mais crítica aquela que tem menos opções em termos de intervalo de tempo em que pode ser executada. Para analisar as opções em intervalo de tempo é utilizado o conceito de janela de demanda definido como o intervalo de tempo entre instante mais cedo em que uma operação pode ser iniciada e o instante mais tarde que esta mesma operação pode ser finalizada. As janelas de demanda de todas as operações em um flow shop podem ser determinadas a partir das relações de precedência e da definição do horizonte de tempo global dentro do qual se pretende obter a solução do problema de scheduling (Anexo C).

Definindo DW_j ("demand window") como a duração do intervalo de tempo em que pode ser executada a operação j, medido na unidade de tempo conveniente ao problema e e_j como a duração da operação j, define-se a criticalidade da operação como a relação $CR_j=e_j/DW_j$. Obviamente $CR_j \leq 1$, e o caso $CR_j=1$, implica em que se tem apenas uma opção para alocar a operação.

A criticalidade de uma operação definida desta forma leva apenas em consideração a demanda, de um dos recursos envolvidos: o processador.

Em (Sycara,1991) este conceito de criticalidade da demanda de um processador por uma operação é modificado de forma a obter-se a função de verosimilhança de que o processador esteja ocupado pela operação j. Calcula-se para cada intervalo unitário de tempo o número de possíveis alocações da operação j em que é exigida a utilização do processador naquele intervalo, e divide-se pelo número total de alocações possíveis.

A Figura 3.6 compara as duas definições para um caso particular em que $DW_j=6$ e $e_j=3$.

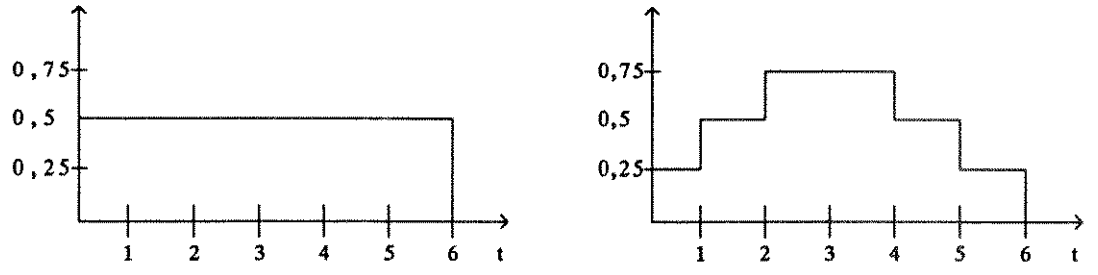


Figura 3.6 - Comparação entre diferentes definições de criticalidade

Conceitos semelhantes a este mas definidos a nível de tarefa ou ordem parecem ter sido utilizados nos sistemas ISIS (Fox,1983) e OPIS (Smith,1985). Posteriormente estes autores propuseram a definição a nível de operação como descrito acima como uma forma de análise mais detalhada (Sadeh,1989). Isto evita a alocação de operações não críticas que seriam alocadas simultâneamente às operações críticas se todas elas pertencessem a uma tarefa ou ordem detectada como crítica. Esta abordagem não coloca nenhuma dificuldade quando se trata de job-shops, como é o caso nos trabalhos citados acima. No caso do flow shop a análise a nível de operação implica em procedimentos específicos para impedir que sucessivas alocações de operações tornem a solução final infactível (Anexo C).

3.5.3.3. Quantificação do caráter crítico de um recurso

As referências (Keng,1985) e (Sycara,1991) têm uma abordagem semelhante para quantificar o caráter crítico de um recurso. A utilização de um determinado recurso r por uma operação i depende de sua execução em um um processador j. O procedimento então é o seguinte:

- Dentro de uma janela de demanda de um processador j pela operação i, quantifica-se a demanda de um recurso r pela criticalidade definida na seção anterior;
- Para cada intervalo de tempo da intersecção das janelas de demanda onde exista demanda pelo recurso r adicionam-se as criticalidades. Este valor é denominado em (Keng,1988) crucialidade do recurso.

Um intervalo de tempo no qual o valor assim calculado seja grande indica um alto grau de competição pelo recurso entre as várias operações envolvidas.

Deve-se notar que no cálculo da crucialidade não figura a quantidade do recurso r solicitado, e portanto a crucialidade não indica naquele intervalo de tempo se a demanda de recurso será efetivamente superior à oferta. Um valor elevado da crucialidade indica apenas que é muito provável que naquele instante existam operações em processadores solicitando o mesmo recurso, o que não necessariamente supõe um conflito. A crucialidade do recurso como definida acima é utilizada nas duas referências citadas para a alocação de operações e processadores no tempo. Isto será discutido na seção 3.5.3.4.

Uma maneira de procurar contornar esta deficiência seria utilizar outra medida do caráter crítico de um recurso. A seguir é apresentada em linhas gerais uma proposta de medida de caráter crítico e como ela poderia ser utilizada para a identificação de prováveis conflitos.

- A criticalidade (normalizada) de um intervalo de tempo como definida na seção anterior é uma medida da probabilidade de que neste intervalo de tempo o processador esteja ocupado pela operação. Portanto indica a probabilidade de que se esteja consumindo um certo nível ρ do recurso r para o processamento da operação (supõe-se um consumo constante do recurso durante toda a operação).

- Em cada intervalo de tempo podem ser executadas simultâneamente várias operações em processadores distintos. Seja ρ_i o consumo do recurso r pela operação i. O produto das criticalidades associadas a este intervalo de tempo a cada operação será a probabilidade de que neste intervalo de tempo o consumo do recurso solicitado seja $\Sigma\rho_i$.

A utilização deste procedimento forneceria uma indicação de possível conflito quando a demanda no intervalo de tempo fosse superior à disponibilidade e, neste caso, a probabilidade seria diferente de zero. O produto das criticalidades implica que o resultado não é nulo apenas na intersecção das janelas, ou seja perde-se a informação de demanda de recurso nos intervalos de tempo onde uma ou mais operações não pode ser executada. Isto pode ser contornado através de um processamento sucessivo das operações em que, a cada passo são evidenciados os intervalos de tempo onde os conflitos podem ocorrer pela coincidência das operações já analisadas, e no fim é feita uma seleção dos conflitos mais críticos.

3.5.3.4. Propostas de técnicas mistas de scheduling

Nesta seção descrevem-se dois sistemas de scheduling descritos na literatura com o objetivo de apresentar suas estruturas. Diversos aspectos importantes de implementação especialmente no que se refere à propagação de restrições e backtracking não são detalhados nestas referências.

A. (Keng,1988)

O sistema, denominado sistema de planejamento sensível às interações, só considera como recursos compartilhados os processadores. São repetidos sucessivamente os quatro passos seguintes:

Passo 1 - Seleção da operação com maior criticalidade. A criticalidade como definida pelos autores é constante para todos os intervalos de tempo da operação.

Passo 2 - Construção de todas as soluções possíveis para alocação da

operação selecionada.Nesta fase determina-se a crucialidade de cada intervalo de tempo.

Passo 3 - Seleção e alocação da operação que corresponde à menor crucialidade. Ou seja, utiliza uma política do tipo "menor impacto", alocando a operação no tempo de forma a diminuir a competição pelo recurso comum com outras operações ainda não alocadas.

Passo 4 - Alocação da operação e propagação das restrições.

Após a alocação da operação, as janelas de tempo das operações não alocadas podem necessitar modificações.

A propagação **intra-ordem** (seção 3.5.3.1) modifica as janelas de demanda das operações dentro da mesma tarefa ou ordem impondo que as relações de precedência sejam satisfeitas.

As propagações **inter-ordem** eliminam os intervalos de tempo em que o processador acaba de ser alocado, das janelas de operações de outras tarefas solicitando o mesmo processador.

Se nesta fase for detectada alguma infactibilidade, por exemplo a não existência de janela com duração suficiente para uma operação, o sistema ativa uma função de supervisão ("contradiction handling").A função de supervisão correponde ao backtracking.

Os autores propõem duas estratégias (heurísticas) para viabilizar a operação prejudicada: eliminar as alocações de operações já feitas na tarefa que contem a operação prejudicada ou a modificação do tempo total (due time) da tarefa. Em seguida o processo descrito nos passos 1 a 4 é repetido.

A propagação inter-ordem como proposta é utilizada apenas para o recurso compartilhado processador. Para outros recursos compartilhados, como por exemplo utilidades, a propagação não pode ser feita como indicado porque a alocação não é exclusiva de uma operação.

As heurísticas que constituem a função de supervisão podem produzir mudanças significativas no scheduling parcial obtido até a detecção da infactibilidade. Certamente isto vai levar a que o sistema não se oriente para soluções próximas àquela que gerou a infactibilidade gerando novas necessidades de backtracking. (Sadeh,1988) indica que podem ser utilizadas heurísticas alternativas, não especificadas na referência, que gerem menos perturbações no scheduling parcial, mantendo o backtracking em um nível aceitável.

B. (Sycara,1991),(Sadeh,1989)

A proposta na primeira referência é desenvolvida para sistemas distribuídos de scheduling, onde cada agente toma decisões de seqüenciamento e alocação sobre um sub-sistema tendo uma informação parcial do resto da planta. A metodologia proposta para a detecção de operações e recursos críticos pode também ser aplicada em um sistema centralizado.

O processo de alocação segue um esquema iterativo semelhante ao descrito na proposta anterior. Para facilitar a comparação será utilizada a mesma decomposição em quatro passos.

Passos 1/2 - Seleção da operação com maior criticalidade. Para cada recurso dispõe-se da adição das criticalidades como indicado em 3.5.3.3. escolhe-se o recurso com criticalidade total maior e, para este recurso, a operação com com maior contribuição na criticalidade total. As duas escolhas são feitas analisando o valor máximo a nível de intervalo unitário de tempo.

Passo 3 - Alocação da operação. O instante de início candidato calcula-se uma estimativa da probabilidade de que, escolhendo este instante como início, não haja violação da restrição sobre o recurso (survivability measure). A estimativa é obtida considerando o número de outras operações que contribuem na demanda deste intervalo de tempo e a sua demanda global.

A heurística "escolha do instante inicial menos constrangedor" aloca o início no intervalo de tempo com maior "survivability". Os autores afirmam que esta heurística reduz a necessidade de backtracing por infactibilidades detectadas no passo 4 mas gera alocações pobres em termos de tempo total necessário.

Nos trabalhos citados são delineadas outras heurísticas que procuram fazer algum tipo de balanceamento entre a escolha anterior e outras características importantes tais como: 1) redução do atraso no tempo de entrega (due date), que pode ter sido aumentado devido a infactibilidades (passo 4), 2) redução de estoques intermediários, 3) alocação uma determinada operação num processador específico (quando existem várias opções) e 4) alocação de uma determinada operação num certo horário do dia. Não são detalhados aspectos de implementação destas heurísticas (que certamente implicam numa interação muito grande com o usuário).

Passo 4 - Propagação de restrições e "backtracking". Como resultado da alocação novas restrições são criadas. O sistema propaga as restrições através de um mecanismo conceitualmente semelhante ao descrito na proposta anterior. Nas referências citadas não são fornecidos detalhes. A detecção de infactibilidades implica em backtracking para um estado de alocação anterior, modificando-se o instante de início da operação atual.

3.5.3.5.Particularização para flow shops

No Anexo C é apresentado um exemplo de aplicação da abordagem descrita em Sycara, adaptando-a para o seqüenciamento em flow shops. Neste exemplo o recurso compartilhado é o processador. Por ser um recurso de demanda unitária durante o período de processamento, a utilização da abordagem proposta não apresenta toda a complexidade de um problema de scheduling. No entanto, algumas conclusões importantes podem ser extraídas deste exemplo de aplicação, e são apresentadas a seguir:

- A utilização de abordagens ao scheduling em flow shop, tais como a proposta por Keng e Sycara, tendem a "individualizar" as operações de cada tarefa, pois as atividades de busca e controle são realizadas sobre a operação. Para preservar as seqüências de permutação é necessário então formular como restrições explícitas as condições de preservação do fluxo de tarefas no shop. Em uma abordagem BAB tradicional aplicada ao seqüenciamento e scheduling em flow shops, decide-se qual a próxima tarefa a ocupar o shop e, posteriormente, é realizada a alocação das diferentes operações de cada tarefa.O exemplo de aplicação detalhado no Anexo C mostra a viabilidade de utilização destas abordagens ao scheduling de flow shops;

no entanto a estratégia de alocação empregada é certamente sub-ótima. Além disso, a individualização das operações é feita a custa de um aumento no número de restrições a serem manipuladas de forma a manter a integridade do fluxo de tarefas. Uma forma de contornar este problema poderia ser a alocação simultânea de todas as operações de uma tarefa.

- As curvas probabilísticas agregadas por recurso não permitem fazer predições acerca de conflitos na demanda por recursos compartilhados, permitem apenas identificar gargalos na oferta de recursos. Esta é certamente uma estratégia útil no planejamento de recursos auxiliando na identificação de casos com condições severas de violação na demanda de recursos. A incorporação de curvas probabilísticas a uma estratégia do tipo BAB para a estimação do limitante inferior precisa ainda ser melhor pesquisada. Um dos problemas a ser resolvido é como incorporar a informação de "conflito potencial" na atividade de busca, de forma a orientar a busca na escolha mais adequada de ramos promissores para a solução abreviada do problema.

No exemplo de aplicação apresentado no Anexo C o recursoprocessador tem uma demanda unitária. Para o caso de recursos cuja oferta não seja unitária, a construção das curvas de demanda com base na criticalidade do recurso não é de modo algum evidente, bem como a informação associada ao valor da criticalidade. A soma de todas as demandas de recurso durante toda a duração das janelas de demanda conduzirá certamente a valores de demanda global superestimados, que podem conduzir a uma falsa interpretação de conflitos.

3.6. Conclusão

Como resultado das idéias apresentadas até aqui, podem ser destacados alguns aspectos relevantes:

a) A dificuldade de utilização da formulação matemática para o scheduling de tarefas em flow shops e a inexistência de heurísticas desenvolvidas especialmente para este fim, restringem a abordagem do problema ao emprego dos métodos de busca.Dentre os diferentes métodos de busca optou-se pela utilização do BAB por várias razões tais como:i)facilidade de preservação da direção do fluxo de tarefas;ii)facilidade de implementação de heurísticas nos nós de forma a permitir a redução do espaço de soluções a ser sondado;iii)facilidade de introduzir mecanismos de interferência externa no processo decisório;

b) A medida de desempenho normalmente utilizada como critério para a otimização é o makespan pois ele reflete um custo indireto do sistema, isto é, quanto mais cedo as tarefas ficarem prontas, menor será seu custo. Os demais custos são normalmente utilizados em níveis hierárquicos de planejamento superiores ao do planejamento de curto prazo, de forma que o makespan é normalmente aceito como um critério de otimização conveniente. Além disso, para o caso de seqüenciamento de tarefas é fácil estabelecer fórmulas de recorrência baseadas nos "completion times" para estimar o makespan.Como já se viu, a dinâmica do BAB implica que em cada nó sejam calculadas duas parcelas: uma referente ao custo consolidado até o nó, e outra referente à estimativa do custo mínimo futuro das tarefas não sequenciadas. A presença de restrições dinâmicas tem como principal reflexo a geração eventual de conflitos temporais que devem ser gerenciados apesar de serem difíceis de serem estimados, e que provocam o crescimento do makespan, de forma que seja caracteriza uma medida regular de desempenho;

c) A maior deficiência atual na aplicação dos métodos de busca ao problema do scheduling em flow shops com limitação na oferta de recursos compartilhados diz respeito à atividade de controle da enumeração, pois ela é a responsável pela estimativa de custo associado aos ramos gerados a partir de cada nó. É desejável o desenvolvimento de uma heurística capaz de prever a expansão mínima que deverá sofrer o makespan estimado, de forma a acomodar progressivamente o perfil consolidado de demanda de cada recurso à medida que as tarefas sejam sequenciadas e as operações alocadas. A principal implicação é que torna-se difícil estabelecer a priori se, no caso da ocorrência da demanda, qual a melhor forma de se resolver o conflito : se fazendo uma alocação não conflitante, o que significa normalmente introduzir atrasos no schedule até que a demanda se iguale à oferta, ou se a melhor estratégia é alterar a seqüência de produção (backtracking).

CAPÍTULO 4 - A ABORDAGEM PROPOSTA

4.1. Introdução

Neste Capítulo é apresentada uma abordagem para a solução do problema do scheduling em flow shops com limitação na oferta de recursos comuns.

Na abordagem proposta recorre-se a um método de enumeração do tipo BAB, com o objetivo de minimizar o makespan, em que o limitante inferior é calculado com base no tempo mínimo necessário para executar as tarefas sem que a oferta de recursos seja violada. O valor deste tempo mínimo é o resultado de uma solução de compromisso entre os tempos necessários para a execução das tarefas sem violar a oferta de recursos compartilhados, e o tempo mínimo necessário para executar as tarefas preservando as restrições de oferta de processadores. Neste sentido, os processadores são igualmente tratados como recursos compartilhados.

Este capítulo apresenta a seguinte organização:

a) Apresentação do princípio em que se baseia a estimativa de tempo;

b) Apresentação dos diferentes algoritmos derivados do princípio apresentado ;

c) Apresentação de um exemplo detalhado resolvido aplicando as técnicas discutidas no capítulo 3 e os algoritmos propostos neste capítulo.

4.2. Princípio da Abordagem Proposta

Nos problemas de scheduling com limitação na oferta de recursos comuns, o perfil temporal de oferta dos recursos é conhecido a priori bem como o consumo temporal de cada tarefa em cada processador. O problema reside então em desenvolver um schedule que satisfaça a restrição de oferta sendo otimizado um critério de desempenho do sistema.

No Capítulo 3 foi discutida a utilização de uma abordagem BAB para a construção de uma solução ótima considerando que haja oferta de recusos compartilhados (Seção 3.5.1.2), cuja eficiência poderia ser comprometida em função da utilização de uma abordagem inadequada para a construção do limitante inferior. O objetivo é o incremento da eficiência através da utilização de uma metodologia de cálculo do limitante levando em conta estas restrições. Como consequencia, a necessidade de backtracking deve ser reduzida.

A oferta de recursos é normalmente uma função do tempo. A restrição de desigualdade (3.1) representa uma restrição de factibilidade do schedule.

$$d^l(t) = \sum_{i \in K} \sum_{j=1}^{M} Q_{ij}^l(t) \leq q^l(t) \qquad (3.1)$$

onde:

Q^l_{ij} = Quantidade de recurso l demandada no instante t para executar a tarefa i no processador j

q^l = Quantidade instantânea ofertada de recurso l

d^l = demanda instantânea do recurso l

A utilização máxima dos recursos ocorrerá quando os perfis de demanda e oferta forem coincidentes,

$$d^l(t) = q^l(t) \quad , \quad \forall\, t$$

e neste caso o período de execução das tarefas, T^l, será mínimo. Nestas condições tem-se para o total de recursos l demandado, D^l, e ofertado, Q^l:

$$D^l = Q^l = \int_{t=0}^{t=T^l} q^l(t)\,dt \qquad (4.1)$$

No caso geral, a quantidade total de recurso disponível em um intervalo de tempo é dada por:

$$Q_0^l = \int_{t=t_1}^{t=t_2} q^l(t)\,dt \qquad (4.2)$$

onde:
t=variável tempo

q^l=oferta instantânea do recurso l

Q_0^l= oferta total do recurso l em um intervalo de tempo compreendido entre t_1 e t_2

Por outro lado, a demanda total do recurso necessário para executar todas as tarefas será:

$$D^l = \sum_{i=1}^{N} \sum_{j=1}^{M} \int_{\tau=o}^{\tau=t_{ij}} Q_{ij}^l(\tau)\,d\tau \qquad (4.3)$$

onde :
D^l= demanda total do recurso l para a execução de todas as tarefas

Q^l_{ij}= demanda instantânea do recurso l pela tarefa i no processador j

t_{ij}= tempo de processamento da tarefa i no processador j

A quantidade D^l é conhecida para cada recurso na medida que todos os parâmetros que intervêm na equação (4.3) são dados do problema.

Uma primeira avaliação do tempo mínimo necessário para a execução de todas as tarefas levando apenas em consideração a demanda e oferta de recursos é dada pelo valor de T^l tal que o volume de oferta de recursos no intervalo $(0,T^l)$ seja igual à demanda pelas tarefas a serem programadas.

$$D^l = \sum_{i=1}^{N} \sum_{j=1}^{M} \int_{\tau=o}^{\tau=t_{ij}} Q_{ij}^l(\tau)d\tau = \int_{t=0}^{t=T^l} q^l(t)dt \qquad (4.4)$$

No caso de existir mais de um recurso comum, o período mínimo global é dado por:

$$T = \max_{1 \leq l \leq R} [T^l] \qquad (4.5)$$

onde R é o número de recursos compartilhados.

O período T será apenas uma primeira estimativa do período necessário para a execução das tarefas. As razões são:

- O volume de recursos disponíveis num intervalo de tempo deve ser suficiente para suprir a demanda, mas as tarefas que utilizam estes recursos devem obedecer a restrições de precedência das operações , bem como a não interrupção destas operações;
- Para a execução de uma tarefa num intervalo de tempo, não basta que o volume de recursos disponíveis seja suficiente para acomodar a demanda global, mas deve ser igualmente suficiente para atender a demanda instantanea , sem interrupção, durante a execução da tarefa.

Estas condições, que serão discutidas com mais detalhes, não estão implícitas no cálculo do período mínimo de tempo para a execução das tarefas. A forma de acomodar esta deficiência será então "dilatar" progressivamente a estimativa do período de execução através da incorporação das condições acima citadas, estimando sucessivamente o período mínimo para a execução das tarefas.

4.2.1. Aplicação da Abordagem Proposta ao Método BAB

A utilização de uma abordagem BAB eficiente para a construção de um schedule usando como critério de otimização o makespan, deve obedecer alguns princípios já apresentados no Capítulo 3 (Secção 3.3.1), os quais são:

1) À medida que o schedule parcial for construído, além de factível, existirá um custo associado a cada nó que deverá permanecer no mínimo constante. A forma de se fazer então a estimativa do valor do makespan em cada nó deve garantir que sob nenhuma condição esta estimativa poderá ser reduzida durante o desenvolvimento do schedule (limitante inferior).
2) A eficiência da abordagem BAB, representada pela necessidade de backtracking, é dependente da qualidade da estimativa do limitante inferior. Normalmente esta eficiência é o resultado de uma extensa atividade de cálculo do makespan baseada em "heurísticas", que permitem, no caso sem restrições, estimar com bastante segurança o valor do makespan.

A combinação destes dois princípios exige uma solução de compromisso entre o esforço computacional necessário para sondar um número elevado de nós e o esforço computacional necessário para realizar estimativas "realistas" do valor do limitante inferior. Isto é, é desejável que o valor do limitante inferior no nó reflita com bastante aproximação a

expectativa do valor do makespan à custa de um esforço computacional aceitável. A seguir serão apresentadas as estratégias empregadas nesta abordagem na busca de uma metodologia de estimação do makespan à luz dos princípios acima citados.

Na Figura 4.1 é feita uma representação do perfil de demanda de um recurso l referente a um nó da árvore de busca, e com base nesta representação serão apresentadas as diferentes heurísticas para estimativa do makespan.

Cada nó da árvore de busca é caracterizado por:
1) K: Conjunto ordenado de tarefas já sequenciadas
2) U: Conjunto de tarefas a sequenciar
3) Distribuição temporal das tarefas K já programadas que satisfaz simultâneamente todas as restrições de factibilidade do problema, a saber:
. Restrições de precedência de forma a preservar a direção do fluxo de tarefas;
. Restrições para a manutenção de seqüências de permutação durante o processamento das operações de cada tarefa;
. Restrições na oferta de recursos comuns.
4) Estimativa de um custo mínimo composto da soma de duas parcelas: uma fixa associada ao conjunto K e uma estimativa associada ao conjunto U.

A cada nó pode ser associado um perfil de utilização de recursos como o indicado na Figura 4.1.

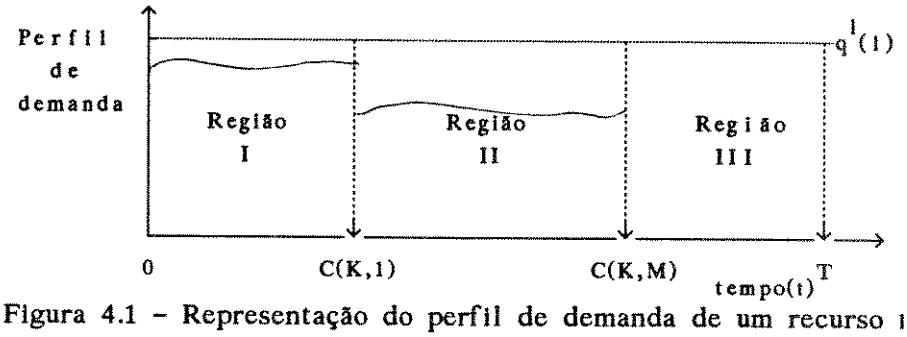


Figura 4.1 - Representação do perfil de demanda de um recurso l em um schedule parcial

Este perfil tem três regiões bem definidas no que se refere à utilização de recursos:

Região I: Esta região está compreendida entre t=0 (início do schedule) e t=C(K,1), onde C(K,1) representa o instante em que a última tarefa da seqüência parcial K é terminada no primeiro processador do shop. Neste intervalo nenhuma tarefa não programada poderá ser iniciada. As restrições de factibilidade estão satisfeitas e os perfis de demanda bem como as programações das tarefas estão consolidadas, isto é, são fixos. A área abaixo da curva de consumo do recurso l, reflete a quantidade deste recurso utilizado no intervalo $0 \le t \le C(K,1)$, denominada aqui de U_I^l. A quantidade de recurso l ofertada no intervalo em questão é:

$$O_I^l = \int_{t=0}^{t=C(K,1)} q^l(t)dt \qquad (4.6)$$

onde:
O_I^l = Quantidade de recurso l ofertada na Região I

A quantidade de recurso efetivamente utilizada, denominada U_I^l, é, em geral, inferior à oferta:

$$U_I^l \leq O_I^l$$

A quantidade U_I^l pode ser calculada por:

$$U_I^l = \sum_{i \in K} \sum_{j=1}^{M} \int_{t=0}^{t=C(K,1)} Q_{ij}^l(t) Z_{ij}(t) dt \qquad (4.7)$$

onde:

$$Z_{ij} = \begin{cases} 1, \text{se a tarefa } i \text{ é executada no processador } j \text{ no instante } t \\ 0, \text{caso contrário} \end{cases}$$

A quantidade de recurso ainda necessária para a execução das operações não realizadas na Região I é igual a: $(D^l - Q_I^l)$.

A não utilização total do perfil de oferta, refletida através do excedente de oferta do recurso l, provoca imediatamente a dilatação do período mínimo necessário para executar todas as tarefas. Esta dilatação pode ser mais facilmente vista comparando o instante em que a última tarefa da seqüência é completada no primeiro processador, C(K,1), e o instante mais cedo que ela seria completada caso fosse utilizado todo o perfil de oferta, $C^{*l}(K,1)$. Este instante é definido para cada recurso l como:

$$U_I^l = \int_{t=o}^{t=C^{*l}(K,1)} q^l(t) dt \qquad (4.8)$$

onde $C^{*l}(K,1)$ é o instante mais cedo que a $K^{\underline{esima}}$ tarefa da seqüência poderia estar terminada no primeiro processador, na hipótese de ser utilizado integralmente o perfil de oferta do recurso l. O valor de $C^*(K,1)$ será dado por:

$$C^*(K,1) = \max_{1 \leq l \leq R} \left[C^{*l}(K,1) \right] \qquad (4.9)$$

onde $C^*(K,1)$ representa o instante mais cedo que a $K^{\underline{esima}}$ tarefa da seqüência poderia estar terminada no primeiro processador de forma a manter o valor de T inalterado.

Obviamente $C(K,1) \geq C^*(K,1)$ porque no cálculo deste último não foram levados em consideração as conseqüências das restrições de precedência.

No caso em que:

$$C(K,1) > C^*(K,1)$$

o valor de T deve ser alterado. O valor final da alteração não depende exclusivamente do desempenho na utilização dos recursos nesta região, mas também nas regiões subsequentes II e III.

Região II: A região II é compreendida entre t=C(K,1) e t=C(K,M), onde C(K,M) é o instante em que a K$^{\underline{esima}}$ tarefa deixa o shop e portanto representa o makespan da seqüência parcial K. No limite inferior desta região (t=C(K,1)) é que poderá ser iniciado o processamento das operações da próxima tarefa candidata a entrar no shop. Com relação ao consumo de recursos existe um perfil de demanda de recursos parcialmente comprometido, cuja capacidade de acomodar de maneira mais ou menos eficiente as diferentes tarefas candidatas a ocupar a posição (K+1) na seqüência parcial precisa ser analisada. A quantidade de recurso i demandada neste intervalo pelas tarefas programadas é:

$$U_{II}^{I}=\sum_{i\in K}\sum_{j=1}^{M}\int_{t=C(K,1)}^{t=C(K,M)} Q_{ij}^{I}(t)Z_{ij}(t)dt \qquad (4.10)$$

onde:

$$Z_{ij}=\begin{cases}1, \text{se a tarefa } i \text{ é executada no processador } j \text{ no instante } t\\ 0, \text{caso contrário}\end{cases}$$

U_{II}^{I}= Quantidade de recurso i utilizada pelas tarefas programadas na Região II

e a quantidade de recursos ofertados na região II é:

$$O_{II}^{I}=\int_{t=C(K,1)}^{t=C(K,M)} q^{I}(t)dt \qquad (4.11)$$

onde:

O_{II}^{I} = Quantidade máxima de recurso i ofertada na Região II

A oferta excedente de recurso na Região II será então:

$$E_{II}^{I}=O_{II}^{I}-U_{II}^{I} \qquad (4.12)$$

Esta região poderá acomodar no máximo (M-1) operações da tarefa candidata a integrar a seqüência parcial, já que o último processador somente estará livre em t=C(K,M). Portanto , apenas a partir deste instante poderá ser iniciada a M$^{\underline{esima}}$ operação da tarefa candidata.

Após a programação de uma tarefa (K+1) é necessário alocar seqüencialmente as operações da tarefa candidata, obedecidos os limites de oferta instantânea dos recursos, o que vai levar a uma redefinição dos limites da Região II. O objetivo no desenvolvimento do schedule é que o perfil de oferta seja utilizado de maneira ótima, de modo que deve ser garantido que o perfil excedente nesta região seja utilizado da melhor maneira possível. A utilização deste excedente não é função exclusivamente da demanda de cada tarefa. De fato, o valor do excedente que poderá ser utilizado por uma tarefa candidata depende dos instantes factíveis de início de cada operação. Chamando (K+1) a tarefa candidata a integrar a

seqüência parcial, o instante real que cada uma das operações poderá ser iniciada no processador correspondente depende de (Anexo A) :

. Instante em que o processador é liberado pela operação equivalente da tarefa anterior, C(K,j)

. Instante em que a operação anterior da mesma tarefa é terminada, C(K+1,j-1)

. Instante mais cedo em que o perfil remanescente de oferta do recurso i é capaz de acomodar o processamento da operação candidata, $C^{i}_{ini}(K+1,j)$.

Região III: A região III é compreendida entre t=C(K,M) e t=T, onde T é o valor estimado do makespan para a execução de todas as tarefas. Nesta região, a oferta total de recursos, compreendida entre t=C(K,M) e T, ou seja o valor estimado do makespan para executar todas as tarefas, será:

$$O^{i}_{III} = \int_{t=C(K,M)}^{t=T} q^{i}(t)dt \qquad (4.13)$$

onde:

O^{i}_{III} = Quantidade total de recurso i ofertada na Região III

Na Região III não há qualquer informação acerca das condições de utilização dos perfis de oferta, na medida que não há ainda operações programadas para t≥C(K,M).

A partir das informações acerca de cada uma das regiões que compõem o diagrama de utilização de recursos de uma seqüência parcial dada, foram desenvolvidos os algoritmos com restrições nos recursos para cálculo do limitante inferior baseados na abordagem Branch and Bound.

4.3. Equações e algoritmos para a utilização da abordagem BAB

A abordagem proposta na seção anterior pode ser aplicada a qualquer método de busca em árvore. Neste trabalho, pelas razões apontadas no Capítulo 3 (Seção 3.5.1.2) foi escolhido um método de busca tipo BAB.

O BAB é desenvolvido através da aplicação sucessiva de duas atividades: busca e controle, e cada nó é identificado por um valor de um limitante inferior (lower bound), que deve permanecer no mínimo constante ao longo de cada caminho. O tempo mínimo necessário para a execução das tarefas, que reflete a estimativa do valor do makespan, será calculado sob a hipótese que, descontada a parcela de utilização de recursos já consolidada, a eventual expansão deste tempo a partir do valor inicial calculado em (4.5), será feita com base na utilização integral do perfil futuro de oferta dos recursos. Esta estratégia reflete um nível máximo de utilização da oferta de recursos, e portanto garante um limitante inferior não decrescente.

O desenvolvimento da atividade de controle do nó exige, em primeiro lugar, a atualização do perfil de utilização de recursos, a fim de delimitar as três regiões apresentadas na seção (4.2).

A atualização do perfil de utilização dos recursos em um nó deve

ser feita de modo a garantir a factibilidade do schedule a qual, por sua vez, depende de um conjunto de restrições, que são:

1- Preservação da direção do fluxo de tarefas no shop, isto é, as operações de cada tarefa devem ser executadas seqüencialmente do processador 1 até o processador M;

2- Durante sua execução, as operações não podem ser interrompidas;

3- O perfil de oferta de recursos compartilhados deve ser capaz de acomodar a nova demanda sem que, a cada instante, seja violada a oferta instantânea de recursos.

As três condições apresentadas acima são utilizadas para definir os instantes factíveis de início das operações de uma determinada tarefa. De acordo com estas restrições, o instante de início de uma operação, que será chamado de $C_{ini}(K,j)$, depende de:

a- Instante em que a operação anterior da mesma tarefa estiver terminada, C(K,j-1).

b- Instante em que o processador exigido para executar a operação em questão estiver liberado, C(K-1,j).

c- Instante mais cedo que o perfil de oferta dos recursos é capaz de acomodar a demanda, respeitados os instantes C(K,j-1) e C(K-1,j).Isto significa que, para $t \geq \max [C(K,j-1),C(K-1,j)]$ deverá ser identificado o instante mais cedo em que haverá um intervalo de tempo ininterrupto de dimensão igual ao tempo de processamento da operação em questão, capaz de acomodar simultâneamente o volume de demanda global de cada um dos recursos necessários, bem como suas demandas instantâneas. Para cada recurso, este instante será chamado $C^{l}_{ini}(K,j)$.

Em resumo, o instante mais cedo factível para o início da operação j da tarefa candidata a ocupar a posição K na seqüência parcial será obtido obedecendo a relação:

$$C_{ini}(K,j)=\max \left\{C(K-1,j),C(K,j-1), \max_{1 \leq l \leq R} [C^{l}_{ini}(K,j)]\right\} \qquad (4.14)$$

onde:
$C_{ini}(K,j)$ = instante de início da tarefa que ocupa a posição K na seqüência no processador j

Após a atividade de controle, é desenvolvida a atividade de busca que consiste na escolha do nó candidato a ser expandido.

Estas atividades de controle e busca são realizadas sucessivamente até a solução final.

Para um caso exclusivamente de seqüenciamento de tarefas em flow shops, esta forma de considerar todos os possíveis ramos de um nó igualmente promissores parece ser adequado, já que todas as operações de cada uma das tarefas candidatas podem ser iniciadas potencialmente no instante em que os processadores são liberados pela última tarefa seqüenciada. A modificação dos instantes de início das operações da tarefa candidata serão devidas exclusivamente ao tempo de processamento de cada

operação. De fato neste caso a relação (4.14) se reduz a:

$$C_{in\,i}(K,j)=\max\left\{C(K-1,j),C(K,j-1)\right\} \qquad (4.15)$$

e no caso particular de j=1 será $C_{in\,i}(K,1)=C(K-1,1)$. Isto é, qualquer que seja a próxima tarefa a ser sequenciada, o instante de início será sempre C(K-1,1). Desta forma, as heurísticas usadas para calcular o makespan partirão do mesmo valor inicial, e a programação efetiva da tarefa candidata a integrar a seqüência parcial será feita a partir deste instante.

A presença de recursos compartilhados pode fazer com que o instante factível de início da primeira operação de uma tarefa candidata não seja C(K-1,1). O atraso no início da primeira operação dado pela diferença $C_{in\,i}(K,1)-C(K-1,1)>0$, poderá refletir em uma dilatação do valor de C(K,M).

Com base nestas observações foram propostos três algoritmos que usam os princípios citados:

. Algoritmos A e B, caracterizados por levar apenas em consideração, para as tarefas não sequenciadas, a demanda e oferta de recursos;

. Algoritmos da classe C onde, para as tarefas não seqüenciadas, o cálculo do limitante inferior leva em consideração as restrições sobre os recursos e os tempos de processamento das operações que as constituem assim como as relações de precedência entre elas.

4.3.1. Algoritmo A

No algoritmo A a idéia básica é supor que toda a oferta futura de recurso (Regiões II e III) será integralmente utilizada para a execução das tarefas remanescentes. De fato, no intervalo t=o até t=C(K,M), a quantidade total de recurso $(U_I + U_{II})$ utilizada é:

$$U_K^l = \sum_{i\in K}\sum_{j=1}^{M}\int_{t=o}^{C(K,M)} Q_{ij}^l(t)Z_{ij}(t)dt \qquad (4.16)$$

onde:
U_K^l= Quantidade total do recurso l utilizada efetivamente nas regiões I e II.

Existe ainda uma demanda de recursos compartilhados não atendida, referente às tarefas não programadas, dada por:

$$Q_U^l=D^l-U_K^l \qquad (4.17)$$

onde:
Q_U^l= Quantidade de recursos necessárias para executar as tarefas não programadas.

A utilização futura dos recursos compartilhados ocorre em seu nível máximo quando:

. A oferta excedente de recurso na região II, E^{l}_{II}, for integralmente utilizada e,

. Na região III, a demanda remanescente, $Q^{l}_{U}-E^{l}_{II}$, for utilizada no limite máximo de oferta.

Desta forma será:

$$(Q^{l}_{U}-E^{l}_{II})=\int_{t=C(K,M)}^{t=LB^{l}_{A}} q^{l}(t)dt \qquad (4.18)$$

expressão que permite calcular o valor do limitante inferior para cada recurso l (o sub-índice A em LB_A refere-se al Algoritmo A).

De acordo com (4.5) o valor do limitante inferior no Algoritmo A será:

$$LB_A = \max_{1\le l\le R}\left\{LB^{l}_{A}\right\} \qquad (4.19)$$

O procedimento apresentado a seguir é o realizado a partir de um nó "pai" e portanto para este nó já são conhecidos:

- A seqüência programada K-1 e portanto C(K-1,1) e C(K-1,M);
- O conjunto de tarefas não sequenciadas, U+1;
- O valor do limitante inferior;
- O perfil atualizado de utilização dos recursos.

Para que este nó tenha sido escolhido para expansão pelo método BAB, é ele que apresenta o limitante inferior mínimo na fila de nós candidatos.

A.1) Para cada tarefa não programada, define-se um nó filho. A programação da tarefa requer a determinação dos instantes de início de cada uma das operações dela com base na equação (4.14). A determinação de $C^{l}_{ini}(K,j)$, $1\le l\le R$, nesta equação implica na compatibilização do perfil de demanda do recurso l na operação no processador j com o perfil de oferta remanescente como já foi descrito na seção anterior.

Como resultado obtém-se C(K,1) e C(K,M). A atividade de controle do BAB será então desenvolvida tendo como base a seqüência parcial K e o conjunto de tarefas não programadas U.

A.2) Para o cálculo do limitante inferior associado ao nó, LB_A, é calculado inicialmente:

U^{l}_{II}, $1\le l\le R$, recurso l utilizado pelas tarefas programadas nas região II (equação 4.10)

U^{l}_{K}, $1\le l\le R$, recurso l utilizado pelas tarefas já programadas nas regiões I e II (equação 4.16)

O^{l}_{II}: oferta de recurso l na região II (equação 4.11)

A.3) Calcula, E^{l}_{II}, a oferta excedente na região II:

$$E_{II}^{l}=\int_{t=C(K,1)}^{t=C(K,M)} q^{l}(t)dt-U_{II}^{l}=O_{II}^{l}-U_{II}^{l} \qquad (4.20)$$

A.4) Calcula a quantidade de recurso necessária na região III, Q_{III}^{l}, através da expressão:

$$Q_{III}^{l}=D^{l}- U_{K}^{l} - E_{II}^{l} \qquad (4.21)$$

A.5) Calcula o limitante inferior do nó filho de forma que:

$$Q_{III}^{l}=\int_{t=C(K,M)}^{t=LB_{A}^{l}} q^{l}(t)dt \qquad (4.22)$$

$$LB_{A}= \max_{1\le l\le R}\left\{LB_{A}^{l}\right\} \qquad (4.19)$$

O limitante inferior calculado através do algoritmo proposto será sempre uma estimativa mínima do tempo necessário para executar todas as tarefas, já que pressupõe a utilização máxima do recurso mais crítico a cada ciclo. Esta condição é garantida pela expressão (4.19) através da qual é escolhido o maior valor de LB_{A}^{l}. Este valor máximo é ditado pelo recurso que atua como gargalo naquele nó específico da árvore.

Deve-se notar que neste algoritmo o valor do limitante, no que se refere à parcela correspondente às tarefas não programadas, é calculado exclusivamente com base na oferta de recursos compartilhados, não se considerando os efeitos sobre o cálculo do limitante inferior dos tempos de processamento.

4.3.2. Algoritmo B

A observação dos perfis de demanda e oferta de recursos na região II, conduz às seguintes conclusões:

1) Quando o excedente de oferta é reduzido, isto é, o perfil excedente não comporta praticamente nenhuma outra tarefa, é provocado imediatamente um aumento da demanda na região III. Como a oferta de recursos é limitada, isto é, o perfil de oferta é fixo, deve haver um aumento no valor de T. O objetivo é atender na região III uma demanda impossível de ser atendida total ou parcialmente na região II.

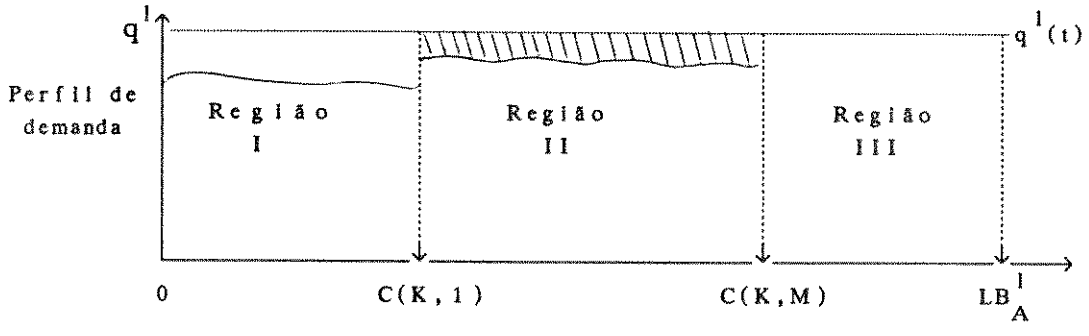
2) O fato de haver um excedente elevado na oferta de recursos na região II, não significa necessariamente que este excedente possa ser efetivamente utilizado para a execução das operações da próxima tarefa candidata. De fato podem ser visualizadas pelo menos duas situações em que a utilização integral deste excedente é difícil embora o valor de E_{II} possa ser elevado:

2a) Há uma folga pequena entre o perfil de demanda e o perfil de oferta, mas o intervalo {C(K,1)-C(K,M)} é grande o suficiente para resultar em um valor de E_{II} elevado. Como resultado prático, esta folga somente poderia ser utilizada de forma eficiente por tarefas cujas operações

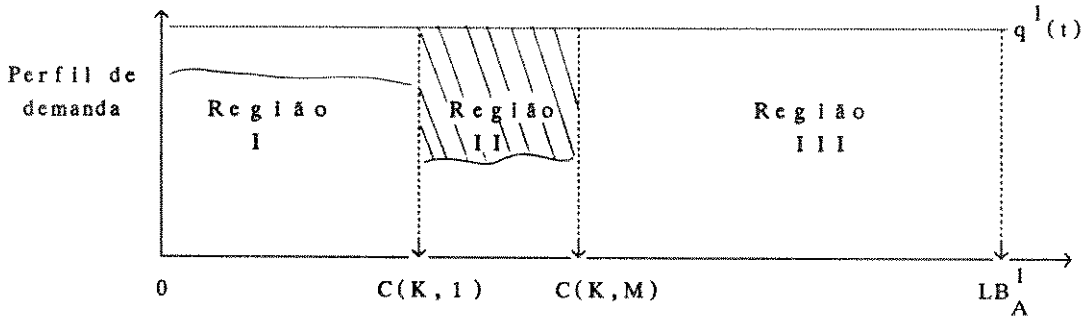
tivessem baixa demanda instantânea $Q_{ij}(t)$. Além disso, a utilização deste excedente está vinculada à continuidade dos intervalos em que ocorrem estas folgas. Em resumo, o excedente somente pode ser utilizado quando as operações tiverem uma demanda instantânea pequena, e os intervalos em que ocorrem sejam factíveis para o processamento das operações sem interrupção.

2b) Há intervalos de tempo curtos em que a folga entre os dois perfis é bastante elevada, de forma que o valor de recurso excedente é elevado. Neste caso pode ocorrer que os intervalos de tempo em que a folga é elevada não sejam suficientes para acomodar a execução de operações.

Estas duas situações estão exemplificadas na Figura 4.2.



a) Intervalo C(K,1) - C(K,M) grande



b) Demanda instantânea pequena na Região II

Figura 4.2- Exemplos de diferentes perfis de folga na Região II

Estas situações sugerem que o conhecimento da natureza do perfil excedente no intervalo C(K,1) e C(K,M) deve ser utilizado de alguma maneira para melhorar a estimativa do limitante inferior.

Seria então desejável que a metodologia de construção do schedule tivesse uma capacidade razoável de prever antecipadamente a possibilidade de conflito na regiao II, melhorando a estimativa do limitante inferior e conseqüentemente reduzindo a freqüência de backtracking.

A importância desta observação e a forma como esta situação pode intervir no processo de construção do limitante inferior podem ser visualizadas na Figura 4.3, onde se representa a situação em que, no que se refere ao recurso I, a primeira operação da tarefa acrescida à seqüência parcial K não pode ser iniciada em C(K,1) porque ou o volume excedente de recursos não é suficiente para atender à demanda, ou porque não é possível atender à demanda instantânea em um ou mais instantes de tempo, somente

podendo ter início em t^{*l}_{K+1} quando a sua demanda de recurso l pode ser acomodada.

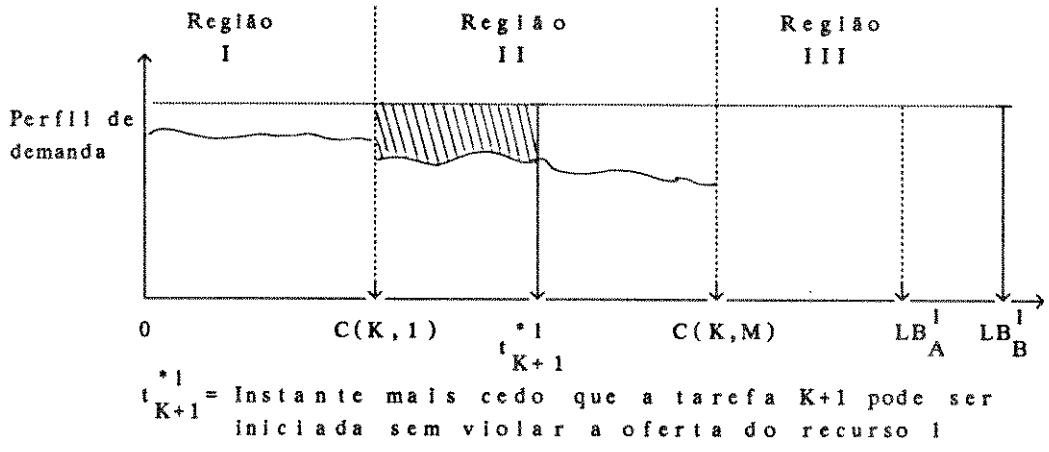


Figura 4.3 - Comparação entre os perfis excedentes de oferta para os Algoritmos A e B

A área hachurada na Região II representa um volume de recursos que não poderá mais ser utilizado o que provoca imediatamente a dilatação do valor de LB^{l}_{A} para LB^{l}_{B}.

No Algoritmo B a idéia básica é utilizar para o cálculo do limitante inferior da seqüência parcial K informações seguras acerca do instante factível de início da primeira operação de cada tarefa candidata a participar da seqüência na posição (K+1). Este procedimento altera parcialmente a estrutura clássica do BAB à medida que cada nó não será caracterizado exclusivamente por um custo mas sim (N-K) custos referentes aos (N-K) ramos possíveis de serem gerados a partir de cada nó da árvore. O valor do limitante inferior associado ao nó K será o mínimo entre os valores de LB calculados para todas as tarefas candidatas a ocupar a posição (K+1) na seqüência.

O percurso entre dois nós de níveis adjacentes, (K) e (K+1), pode ser visto como um conjunto de decisões de alocação seqüencial das operações de uma tarefa candidata a fazer parte da seqüência na posição K+1. Isto significa que a alocação da primeira operação de uma tarefa candidata influencia o instante mais cedo em que a segunda tarefa pode ser iniciada, e assim sucessivamente. Ao ser programada a M^{esima} operação da tarefa candidata a ocupar a posição (K+1) da seqüência, ter-se-á atingido o nível (K+1).

A proposta no Algoritmo B não é percorrer integralmente o caminho entre dois nós adjacentes através da alocação seqüencial de todas as operações da tarefa candidata. Trata-se unicamente de inferir a influência sobre o valor do makespan do fato que a primeira operação da tarefa candidata a ocupar a posição (K+1) na seqüência parcial,não possa ser iniciada no instante C(K,1).

Para o Cálculo do limitante inferior associado a um nó relativo à

seqüência parcial K, consideram-se a primeira operação de cada tarefa candidata a ocupar a posição K+1 na seqüência. O instante exato, $t^{l}_{K+1,m}$ m $\in$ (N-K), em que a primeira operação, e apenas ela, de uma próxima tarefa poderá ser iniciada, definirá a parcela máxima do perfil excedente do recurso l na Região II que poderá ser utilizada. Não basta identificar um instante qualquer factível, mas sim o instante mais cedo de forma a garantir que a estimativa do makespan ao longo de um determinado caminho não seja reduzida. Neste caso a utilização de recursos na Região II dada pela equação (4.10) e o excedente dado pela equação (4.20) são dados respectivamente pelas equações (4.23a) e (4.23b):

$$U^{l}_{II,K+1,m} = \sum_{i\in K} \sum_{j=1}^{M} \int_{t=t^{l*}_{K+1}}^{t=C(K,M)} Q^{l}_{ij}(t)\, Z_{ij}(t)\, dt \qquad (4.23a)$$

$$E^{l}_{II,K+1,m} = \int_{t=t^{l*}_{K+1}}^{t=C(K,M)} q^{l}(t)\, dt - U^{l}_{II,K+1,m} \qquad (4.23b)$$

A heurística proposta de vinculação do cálculo do limitante inferior ao instante efetivo de início da primeira operação da tarefa associada a um ramo de um nó candidato, ainda conduz a uma estimativa conservadora do valor do limitante. A definição do instante real de início da primeira operação do próxima tarefa permite definir o máximo que poderá ser aproveitado do perfil excedente, em um ramo específico, pois supõe-se que, para $t \geq t^{l*}_{K+1,m}$, o perfil de oferta será integralmente aproveitado.

O cálculo da folga na oferta de recursos na região II pode ser ainda refinado. Na medida que é conhecido o instante mais cedo que a primeira operação da tarefa (K+1) poderá ser iniciada, $t^{l*}_{K+1,m}$, é também conhecido o volume de recursos exigidos para a execução desta tarefa no intervalo $t^{l*}_{K+1,m} \leq t \leq t^{l*}_{K+1,m} + t(m,1)$, correspondente ao intervalo de execução da primeira operação da tarefa candidata.

Desta forma, a quantidade de recurso efetivamente comprometida nas regiões I e II pela seqüência parcial K e a primeira operação da tarefa candidata, denominado $U^{l*}_{K+1,m}$, será dado por:

$$U^{l*}_{K+1,m} = \left\{ \sum_{i\in K} \sum_{j=1}^{M} \int_{t=o}^{t=C(K,M)} Q^{l}_{ij}(t)\, Z_{ij}(t)\, dt \right\} + \int_{t=t^{*l}_{K+1}}^{t=C^{l}(K+1,1)} Q^{l}_{m,1}(t)\, dt \qquad (4.24a)$$

ou ainda:

$$U^{l*}_{K+1,m} = U^{l}_{K} + \int_{t=t^{*l}_{K+1}}^{t=C^{l}(K+1,1)} Q^{l}_{m,1}(t)\, dt \qquad (4.25)$$

Definindo $C^l(K+1,1) = t^{l*}_{K+1,m} + t(m,1)$, a quantidade efetiva de recurso excedente na região II será:

$$E^{l}_{II,K+1,m} = \int_{t=C^{l}(K+1,1)}^{t=C(K,M)} q^{l}(t)\, dt - \sum_{i\in K}\sum_{j=1}^{M} \int_{t=C^{l}(K+1,1)}^{t=C(K,M)} Q^{l}_{ij}(t)\, Z_{ij}(t)\, dt \qquad (4.24b)$$

onde $C^l(K+1,1)$ corresponde ao instante mais cedo que a primeira operação da tarefa candidata m candidata a ocupar a posição (K+1) poderia estar terminada.

Cada tarefa candidata a ser acrescida à seqüência parcial K originará um valor diferente de recursos utilizados (equações 4.23a ou 4.24a) e de recursos excedentes (equações 4.23b e 4.24b). O limitante inferior da seqüência K no algoritmo B, LB_B, é calculado com base no valor mínimo do acréscimos resultante da diminuição dos recursos excedentes. Em outras palavras, LB_B difere de LB_A no acréscimo mínimo descrito acima.

Os principais passos do Algoritmo B para o cálculo do limitante inferior associado a cada um dos ramos são apresentados a seguir.

O procedimento apresentado a seguir é o realizado a partir de um nó "pai" e portanto para este nó já são conhecidos:

- A seqüência programada K-1 e portanto C(K-1,1) e C(K-1,M);
- O conjunto de tarefas não sequenciadas, U+1;
- O valor do limitante inferior;
- O perfil atualizado de utilização dos recursos.

Para que este nó tenha sido escolhido para expansão pelo método BAB, é ele que apresenta o limitante inferior mínimo na fila de nós candidatos.

B.1) Para cada tarefa não programada no nó (K-1), define-se um nó filho. A programação da tarefa requer a determinação dos instantes de início de cada uma das operações dela com base na equação (4.14). A determinação de $C^{l}_{in\,i}(K,j)$, $1 \le l \le R$, nesta equação implica na compatibilização do perfil de demanda do recurso l na operação no processador j com o perfil de oferta remanescente como já foi descrito na seção anterior.

Como resultado obtém-se C(K,1) e C(K,M).

B.2) Para o cálculo do limitante inferior associado ao nó, LB_B, é

calculado inicialmente:

U^{I}_{II}, $1 \leq I \leq R$, recurso I utilizado pelas tarefas já programadas na região II (equação 4.10)

$U^{I}_{K+1,m}$, $1 \leq I \leq R$, recurso I utilizado pelas tarefas já programadas na região II (equação 4.23a ou 4.24a)

O^{I}_{II}: oferta de recurso I na região II (equação 4.11)

B.3) Calcula $E^{I*}_{II,K+1,m}$ através das expressões (4.23b) ou (4.24b).

B.4) Calcula a quantidade de recurso necessária na região III, $Q^{I}_{III,K+1,m}$, através da expressão:

$$Q^{I}_{III,K+1,m} = D^{I} - U^{I}_{K+1,m} - E^{I*}_{II,K+1,m} \qquad (4.26)$$

onde:

$Q^{I}_{III,K+1,m}$ =Quantidade de recurso necessária naregiãoIII,associada a cada ramo de expansão do nó m.

B.5) Calcular o limitante inferior de cada recurso, $LB^{I}_{B,K+1,m}$ associado a cada ramo de expansão do nó (m) através da expressão:

$$Q^{I}_{III,K+1,m} = \int_{t=C(K,M)}^{t=LB^{I}_{B,K+1,m}} q^{I}(t)\, dt \qquad (4.27)$$

B.6) Calcular o limitante parcial relativo à tarefa candidata através da relação:

$$LB_{B,K+1,m} = \max_{1 \leq I \leq R} \left\{ LB^{I}_{B,K+1,m} \right\} \qquad (4.28)$$

B.7) A estimativa do makespan no nó K será então:

$$LB_{B} = \min_{m \in N-K} \left\{ LB_{B,K+1,m} \right\} \qquad (4.29)$$

Na Figura 4.3 é apresentada uma comparação gráfica entre as áreas consideradas "úteis" nos Algoritmos A e B. No caso do Algoritmo A toda a oferta excedente de recurso na região II poderá, em princípio, ser utilizada. A duração da região III será determinada exclusivamente pela oferta não utilizada na região II. No caso do Algoritmo B, é associado um custo mínimo a cada um dos ramos de um nó, o qual, na abordagem proposta, é função do instante de início da primeira operação da próxima tarefa candidata. Esta estratégia implica que poderá haver uma parcela não utilizada dos recursos disponíveis na região II, representada pela área hachurada na Figura 4.3 de forma que o volume de recursos não utilizados se reflete em um aumento do tempo necessário para executar as tarefas não programadas.

4.3.3. Algoritmo C

Os Algoritmos A e B propostos anteriormente baseiam o cálculo do limitante inferior exclusivamente no período ainda necessário para que a oferta de recursos seja suficiente para executar as tarefas remanescentes.No entanto não foi considerada a hipótese que este período remanescente compreendido entre t=C(K+1) e $t=LB_A$ no caso do Algoritmo A e $t=t^{*1}_{K+1,m}$ e $t=LB_B$ no caso do algoritmo B, seja suficiente para executar as tarefas não programadas em função dos tempos de processamento e da oferta limitada dos processadores.

Podem existir situações em que a influência sobre o makespan do tempo ainda necessário para executar as tarefas seja maior do que a estimativa de tempo necessário para dispor da quantidade de recursos requerida. Deve-se notar que os tempos de processamento, a disponibilidade de processadores e as restrições de precedências são levadas em consideração na alocação das operações na região II (eq. 4.14). Ou seja,o limitante inferior calculado seja no algoritmo A ou B incorpora estes fatores para as tarefas já programadas, porém estes dois algoritmos não consideram estes fatores no cálculo da duração da região III, que por sua vez influencia o limitante inferior LB.

Uma forma de incorporar aos dois algoritmos a importância relativa que as restrições impostas pela estrutura de processamento possam vir a ter na construção do schedule é através do cálculo do limitante a partir de duas quantidades: uma baseada no período necessário para não violar o perfil de oferta de recursos compartilhados, e o outra baseada em heurísticas que estimem o tempo necessário para executar tarefas não sequenciadas, tais como as apresentadas na Tabela 3.5. O valor do limitante resultará da escolha da maior estimativa de tempo.

Foi utilizada a heurística "Full Machine Based Bound" (FMBB) para compor os Algoritmos da categoria C. De acordo com esta heurística o instante mais cedo que cada uma das operações de uma tarefa não programada poderá ser iniciada é C(K,j), e a este valor são adicionados os tempos globais mínimos de processamento baseados na hipótese dos processadores estarem sendo continuamente utilizados.

Foram propostos dois Algoritmos da categoria C : o primeiro como uma modificação do Algoritmo A e o outro como uma modificação do Algoritmo B.

Algoritmo C_1

Os principais passos do Algoritmo C_1 para o cálculo do limitante inferior associado a um nó da árvore são:

O procedimento apresentado a seguir é o realizado a partir de um nó "pai" e portanto para este nó já são conhecidos:

- A seqüência programada K-1 e portanto C(K-1,1) e C(K-1,M);
- O conjunto de tarefas não sequenciadas, U+1;
- O valor do limitante inferior;
- O perfil atualizado de utilização dos recursos.

Para que este nó tenha sido escolhido para expansão pelo método

BAB, é ele que apresenta o limitante inferior mínimo na fila de nós candidatos.

C_1.1) Para cada tarefa não programada, define-se um nó filho. A programação da tarefa requer a determinação dos instantes de início de cada uma das operações dela com base na equação (4.14). A determinação de $C^l_{in\,l}(K,j)$, $1 \le l \le R$, nesta equação implica na compatibilização do perfil de demanda do recurso l na operação no processador j com o perfil de oferta remanescente como já foi descrito na seção anterior.

Como resultado obtém-se C(K,1) e C(K,M). A atividade de controle do BAB será então desenvolvida tendo como base a seqüência parcial K e o conjunto de tarefas não programadas U.

C_1.2) Para o cálculo do limitante inferior associado ao nó, LB_{C_1}, são calculados inicialmente:

U^l_{II}, $1 \le l \le R$, recurso l utilizado pelas tarefas programadas nas regiões I e II (equação 4.10)

U^l, $1 \le l \le R$, recurso l utilizado pelas tarefas já programadas na região II (equação 4.16)

O^l_{II}: oferta de recurso l na região II (equação 4.11)

C_1.3) Calcular, E^l_{II}, a oferta excedente na região II:

$$E^l_{II} = \int_{t=C(K,1)}^{t=C(K,M)} q^l(t)dt - U^l_{II} = O^l_{II} - U^l_{II} \qquad (4.20)$$

C_1.4) Calcular a quantidade de recurso necessária na região III, Q^l_{III}, através da expressão:

$$Q^l_{III} = D^l - U^l_{K+1} - E^l_{II} \qquad (4.21)$$

C_1.5) Calcular o limitante inferior do nó filho de forma que:

$$Q^l_{III} = \int_{t=C(K,M)}^{t=LB^l_A} q^l(t)dt \qquad (4.22)$$

então será:

$$LB_A = \max_{1 \le l \le R} \left\{ LB^l_A \right\} \qquad (4.19)$$

C_1.8) Calcular o limitante inferior com base na heurística FMBB apresentada na Tabela 3.7, de modo que, o limitante inferior calculado será:

$$LB_{FMBB} = \max_{1 \le j \le M} \left\{ B_j \right\} \qquad (4.30)$$

O cálculo de (4.30) está apresentado na seção (3.4.3.3)

C_1.9) O limitante inferior global será:

$$LB_{C_1} = \max \{ LB_A , LB_{FMBB} \} \quad (4.31)$$

Algoritmo C_2

O procedimento apresentado a seguir é o realizado a partir de um nó "pai" e portanto para este nó já são conhecidos:

- A seqüência programada K-1 e portanto C(K-1,1) e C(K-1,M);
- O conjunto de tarefas não sequenciadas, U+1;
- O valor do limitante inferior;
- O perfil atualizado de utilização dos recursos.

Para que este nó tenha sido escolhido para expansão pelo método BAB, é ele que apresenta o limitante inferior mínimo na fila de nós candidatos.

C_2.1) Para cada tarefa não programada no nó (K-1), define-se um nó filho. A programação da tarefa requer a determinação dos instantes de início de cada uma das operações dela com base na equação (4.14). A determinação de $C^l_{ini}(K,j)$, $1 \leq l \leq R$, nesta equação implica na compatibilização do perfil de demanda do recurso l na operação no processador j com o perfil de oferta remanescente como já foi descrito na seção anterior.

Como resultado obtém-se C(K,1) e C(K,M).

C_2.2) Para o cálculo do limitante inferior associado ao nó, LB_B, é calculado inicialmente:

U^l_{II}, $1 \leq l \leq R$, recurso l utilizado pelas tarefas já programadas na região II (equação 4.10)

$U^l_{K+1,m}$, $1 \leq l \leq R$, recurso l utilizado pelas tarefas já programadas na região II (equação 4.23a ou 4.24a)

O^l_{II}: oferta de recurso l na região II (equação 4.11)

C_2.3) Calcula $E^{l*}_{II,K+1,m}$ através das expressões (4.23b) ou (4.24b).

C_2.4) Calcula a quantidade de recurso necessária na região III, $Q^l_{III,K+1,m}$, através da expressão:

$$Q^l_{III,K+1,m} = D^l - U^l_{K+1,m} - E^{l*}_{II,K+1,m} \quad (4.26)$$

onde:

$Q^l_{III,K+1,m}$ = Quantidade de recurso necessária na região III, associada a cada ramo de expansão do nó m.

C_2.5) Calcular o limitante inferior de cada recurso, $LB^l_{B,K+1,m}$ associado a cada ramo de expansão do nó (m) através da expressão:

$$Q^{l}_{III,K+1,m} = \int_{t=C(K,M)}^{t=LB^{l}_{B,K+1,m}} q^{l}(t)\, dt \qquad (4.27)$$

C_2.6) Calcular o limitante parcial relativo à tarefa candidata através da relação:

$$LB_{B,K+1,m} = \max_{1 \le l \le R} \left\{ LB^{l}_{B,K+1,m} \right\} \qquad (4.28)$$

C_2.7) A estimativa do makespan no nó K será então:

$$LB_B = \min_{m \in N-K} \left\{ LB_{B,K+1,m} \right\} \qquad (4.29)$$

C_2.8) Calcular o limitante inferior com base na heurística FMBB modificada apresentada a seguir:

$$t^{*}_{K+1,m} = \max_{1 \le l \le R} \left\{ t^{*l}_{K+1,m} \right\}$$

$$B_{j,K+1,m} = C(K+1,j) + \sum_{i \in U} t_{ij} + \min_{i \in U} \left\{ \sum_{k=j+1}^{M} t_{ik} \right\} \quad 1 \le j \le M-1$$

$$B_{M,K+1,m} = C(K+1,M) + \sum_{i \in U} t_{iM}$$

com :

$$C(K+1,j) = \max \left\{ t^{*}_{K+1,m} + \sum_{n=1}^{j} t(m,n) \; , C(K, j-1) \right\} \qquad (4.32)$$

O limitante inferior será:

$$LB_{H,K+1,m} = \max_{1 \le j \le M} \left\{ B_{j,K+1,m} \right\}$$

$$LB_H = \min_{m \in N-K} \left\{ LB_{H,K+1,m} \right\} \qquad (4.33)$$

Deve-se notar que a expressão (4.32) é diferente da expressão original da Heurística FMBB, pois o cálculo do tempo mínimo necessário para executar as tarefas não programadas depende do instante em que efetivamente o processador é liberado. Nesta metodologia este instante é denominado t^{*}_{K+1}.

C_2.9) O limitante inferior será:

$$LB_{C_2} = \max \{ LB_B, LB_H \} \qquad (4.34)$$

4.4. Exemplo de Aplicação

Nesta seção será apresentado um exemplo de aplicação da abordagem proposta através do emprego dos algoritmos apresentados anteriormente para o scheduling de tarefas em um flow shop com limitação em recursos comuns, adotando o critério do makespan.

A abordagem proposta será comparada com as propostas de solução apresentadas no Capítulo 3. Neste caso, o problema será resolvido através do emprego de dois caminhos já apresentados:

1) Método BAB Best First em que é gerada uma seqüência de tarefas considerando que apenas os processadores são compartilhados e os demais recursos comuns possuem oferta ilimitada. Após a geração da seqüência, é desenvolvido o perfil de demanda de recursos comuns e as eventuais infactibilidades são resolvidas através da introdução da atrasos na execução das tarefas, de modo a tornar os perfis de demanda factíveis. Esta abordagem será chamada de "Algoritmo D - Compatibilização a posteriori".

2) Método BAB Best First em que o perfil de consumo é compatibilizado em cada nó da árvore. Este método será denominado "Algoritmo E", e consiste em um método BAB clássico em que o perfil de consumo de recurso compartilhado é compatibilizado em cada nó, isto é, o perfil de demanda de recurso da seqüência parcial no nó é factível. No entanto a estimativa do valor do makespan, no que se refere às tarefas não programadas, é feita considerando que apenas os processadores são compartilhados.

4.4.1. Dados do flow shop

O exemplo de aplicação consiste em um flow shop com três processadores onde devem ser executadas quatro tarefas. As principais hipóteses do problema são:

1) O processamento das tarefas ocorre exclusivamente em batelada;
2). Em t=0 todos os processadores estão desocupados e prontos para iniciar o processamento;
3) Não há limitação no prazo de término das tarefas;
4) Não há restrição de precedência entre as tarefas;
6) A armazenagem intermediária é infinita;
7) Os tempos de transferência e preparação dos equipamentos são independentes da seqüência e estão incluídos nos tempos de processamento, e os dados de tempo de processamento estão apresentados na Tabela 4.1;
8) Existe apenas um recurso compartilhado que é oferecido segundo um perfil temporal constante, sendo q(t)=10 unidades;
9) A demanda temporal de recurso compartilhado é constante durante a execução de cada operação de cada tarefa. Os dados de consumo estão apresentados na Tabela 4.2.

Tabela 4.1 - Dados de tempos de processamento

Processador	Tarefa			
	A	B	C	D
1	8	6	7	4
2	9	7	7	7
3	5	7	12	12

Tabela 4.2 - Dados de consumo de recurso compartilhado

Processador	Tarefa			
	A	B	C	D
1	8	6	7	3
2	6	2	3	4
3	4	4	6	8

4.4.1.Solução do exemplo através do Algoritmo D - Compatibilização a posteriori

O algoritmo em que se utiliza a metodologia de compatibilização a posteriori é desenvolvido em duas etapas. Na primeira etapa é gerada uma seqüência de tarefas sendo ignoradas as eventuais infactibilidades do schedule devido a limitação na oferta de recursos comuns. Na segunda etapa,é construído o perfil temporal de demanda de cada um dos recursos compartilhados. Os eventuais conflitos gerados pela violação da oferta de recursos são resolvidos usando-se alguma heurística tais como as regras de prioridade do evento ou do produto, citadas no Anexo B. Esta abordagem significa então a introdução de atrasos na execução das operações das tarefas, que podem resultar no crescimento do valor do makespan. Este crescimento não é controlado e não pode ser predito.

No exemplo em questão, na primeira etapa foi gerada uma seqüência utilizando um BAB Best First juntamente com a heurística FMBB apresentada na Tabela 3.5. Neste caso as hipóteses 8 e 9 deixam de ser aplicadas, sendo substituídas pela hipótese 8 :

8) A oferta de recursos compartilhados é ilimitada.

A seqüência de nós sondados e o perfil de demanda de recursos estão apresentados na Tabela 4.3 e na Figura 4.4 respectivamente. Como se pode observar, existem diversos períodos em que a demanda excede as 10 unidades de recurso disponíveis infactibilizando o schedule. A segunda etapa consiste então em factibilizar o schedule, mantendo a seqüência de tarefas inalterada. Neste exemplo foi utilizada a estratégia de prioridade do evento (Anexo B). O schedule final compatibilizado está apresentado na Figura 4.5.

Comparando-se as Figuras 4.4 e 4.5 observa-se que o makespan cresceu de 47 unidades de tempo para 72 unidades de tempo.

Neste procedimento não há a preocupação de desenvolver schedules ótimos, de forma que, uma vez que o schedule seja factível, não é utilizada qualquer metodologia de busca de uma solução alternativa melhor em termos de makespan.

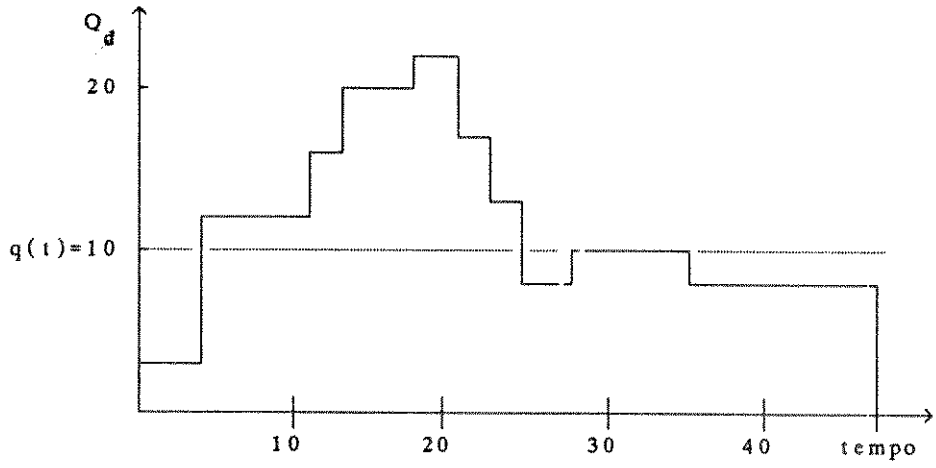


Figura 4.4 - Perfil de consumo - Oferta Ilimitada

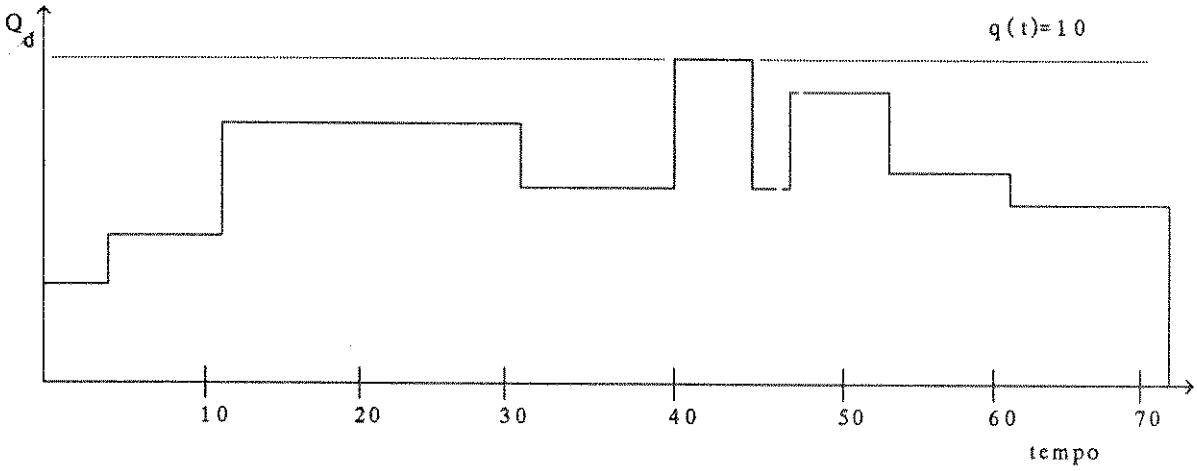


Figura 4.5 - Perfil de demanda após compatibilização

Tabela 4.3 - Seqüência de nós sondados

Seqüência Parcial	Makespan estimado
A	53
B	49
C	50
D	47
DA	47
DB	47
DC	47
DAB	47
DAC	47
DABC	47

Nas Figuras 4.5a e 4.5b são apresentadas respectivamente as curvas do perfil de demanda do recurso compartilhado e a carta de Gantt correspondente à seqüência ótima ABCD.

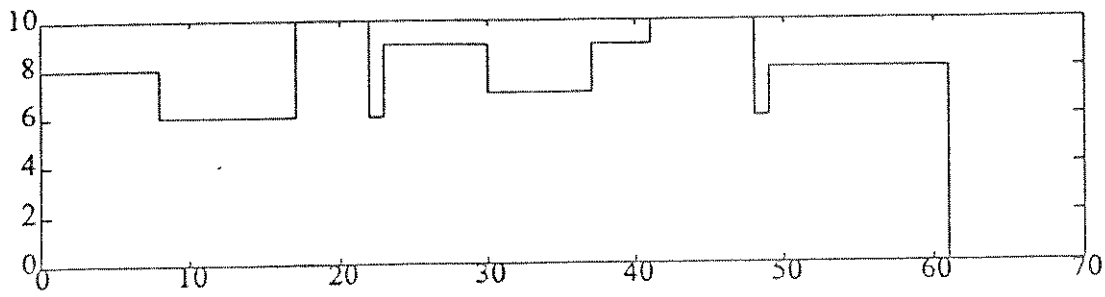


Figura 4.5a - Perfil de demanda do recurso para a seqüência ABCD

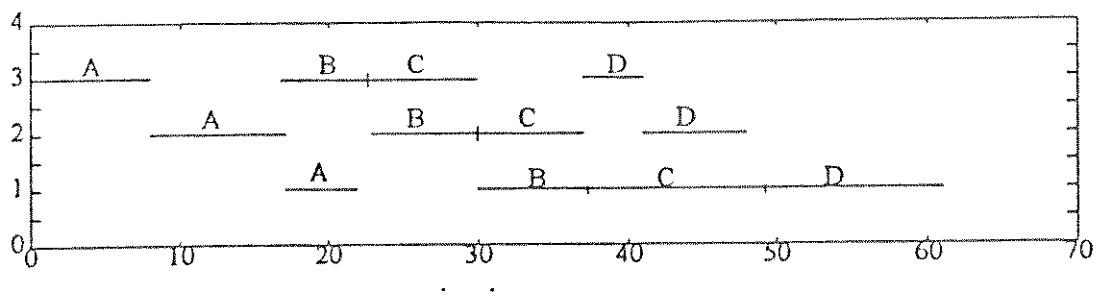


Figura 4.5b - Carta de Gantt para a seqüência ABCD

4.4.3. Solução do exemplo via Algoritmo E- BAB compatibilizado

Uma maneira de construir schedules ótimos, isto é, minimizando um critério de desempenho adotado, é utilizar uma abordagem BAB de forma que, a cada passo do algoritmo, seja garantida a factibilidade do perfil de demanda. A eficiência desta abordagem depende da qualidade da estimativa do custo associado a cada nó. Não se leva em conta o efeito futuro da oferta limitada dos recursos compartilhados, faz-se uma estimativa desta parcela com base em heurísticas tais como as apresentadas na Tabela 3.5.

Neste trabalho foi utilizado um BAB Best First, utilizando a heurística FMBB para estimar o makespan em cada nó. As hipóteses do problema são as hipóteses 1 a 9 apresentadas anteriormente. O algoritmo de alocação e estimativa do makespan que foi inserido na BAB é apresentado a seguir:

Algoritmo E

O procedimento no Algoritmo E é feito a partir de um nó "pai" e portanto para este nó são conhecidos:

- A seqüência programada K-1 e portanto C(K-1,1) e C(K-1,M);
- O conjunto de tarefas não programadas, U+1
- O valor do limitante inferior;
- Perfil atualizado de utilização de recursos.

Para que este nó tenha sido escolhido para expansão pelo método BAB, é ele que apresenta o limitante inferior mínimo da fila de nós candidatos.

E.1) Para cada tarefa não programada, define-se um nó filho. A programação da tarefa requer a determinação dos instantes de início de cada uma das operações dela com base na equação (4.14). A deteminação de $C^{l}_{ini}(K,j)$, $1 \leq l \leq R$, nesta equação implica na compatibilização do perfil de demanda do recurso l da operação no processador j, com o perfil de oferta remanescente.

Com resultado obtém-se C(K,1) e C(K,M). A atividade de controle do BAB será desenvolvida tendo como base a seqüência parcial K e o conjunto de tarefas não programadas.

E.2) O cálculo do limitante inferior para a seqüência K é feito com base na heurística FMBB apresentada na Tabela 3.5, de forma que o seu valor, LB_E será:

$$LB_E = \max_{1 \leq j \leq M} \left\{ B_j \right\} \qquad (4.35)$$

Na Figura 4.6 é apresentada a árvore de nós sondados.

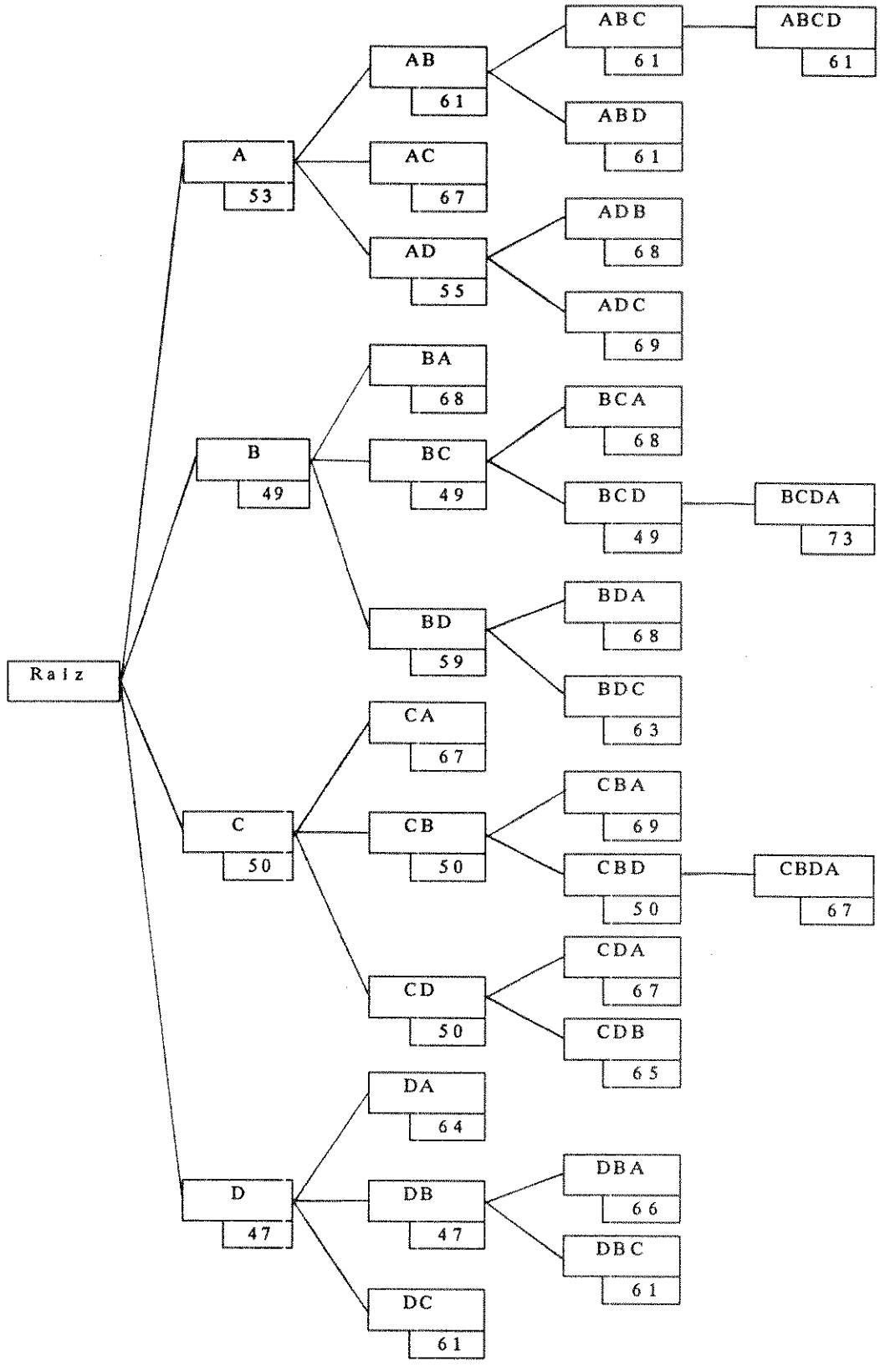


Figura 4.6 - Árvore de nós sondados - Algoritmo E

4.4.4. Solução do exemplo através do uso dos algoritmos propostos (A,B,C_1,C_2)

Para a utilização dos Algoritmos A,B,C_1 e C_2 é necessário calcular inicialmente o valor do período mínimo, T, necessário para a execução de todas as tarefas considerando que a oferta de recursos será integralmente utilizada.

Nas condições apontadas no exemplo se terá (o superscrito i de identificação do recurso será suprimido na medida que existe apenas um recurso compartilhado):

$$D = \sum_{i=1}^{N} \sum_{j=1}^{M} Q_{ij} t_{ij} = 494 \text{ unidades de recurso}$$

$$Q = \int_{t=0}^{t=T} q(t)\, dt = q(T-0) = qT$$

O período mínimo necessário para a execução das tarefas sem violar a oferta de recurso será obtido de :

$$D = Q \rightarrow T = \frac{Q}{q} = \frac{494}{10} = 49{,}4 \text{ unidades de tempo}$$

4.4.4.1. Algoritmo A

A expressão do limitante inferior será simplificada dado que a oferta e a demanda das operações são constantes.

Combinando as equações (4.20), (4.21), e (4.22) será:

$$\int_{t=C(K,M)}^{t=LB_A} q(t)\, dt = D - (U_I + U_{II}) - O_{II} + U_{II}$$

De acordo com (4.11):

$$\int_{t=C(K,M)}^{t=LB_A} q(t) = D - \int_{t=C(K,1)}^{t=C(K,M)} q(t)\, dt - U_I$$

Tendo em conta que q(t) é constante em todo intervalo de tempo, a expressão anterior pode ser escrita como:

$$q\ [LB_A - C(K,M)] + q\ [C(K,M) - C(K,1)] = D - U_I$$

O limitante inferior então é dado por:

$$LB_A = C(K,1) + \frac{D - U_I}{q} \qquad (4.36)$$

onde :

D=Quantidade de recurso disponível

U_I= Quantidade de recurso utilizado na região I

q= Quantidade instantânea de recurso disponível

Na Figura 4.7 é apresentada a árvore de nós sondados.

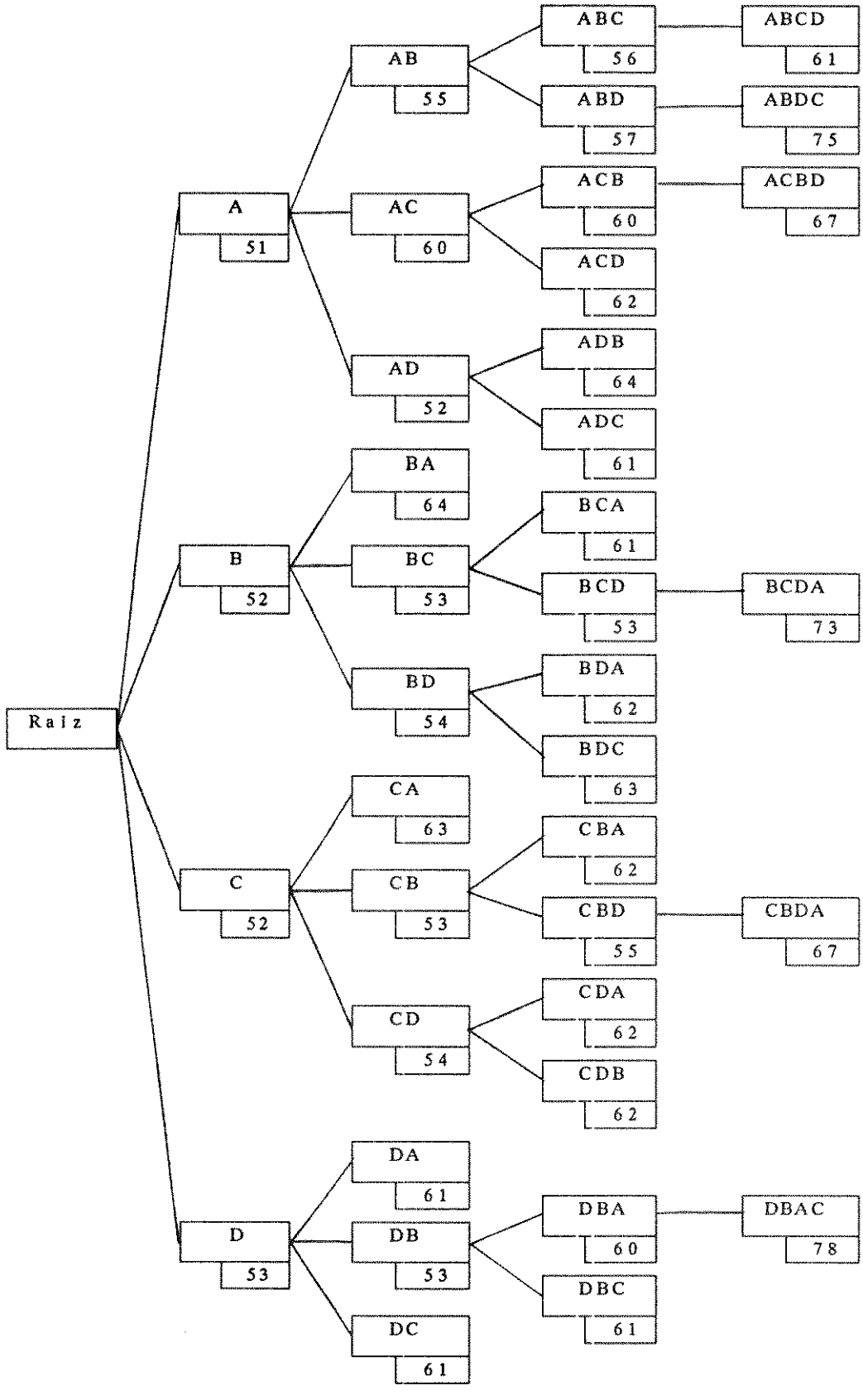


Figura 4.9 -Árvore de busca Algoritmo A

4.4.4.2. Algoritmo B

A obtenção da expressão que permite calcular o valor do limitante inferior para cada ramo candidato a expansão à partir de um nó "pai" correspondente a cada uma das m tarefas não programadas no nó K (m ∈ N-K) cial (K+1) é feita a partir das expressões (4.24a), (4.24b), (4.26), (4.27) e tendo em conta que q(l) é constante, será:

$$q\left(LB_{B,K+1,m}-C(K,M)\right)=D-U^{*}_{K+1,m}-q\left(C(K,M)-C(K+1,1)\right)+\sum_{i\in K}\sum_{j=1}^{M}\int_{t=C(K+1,1)}^{t=C(K,M)}Q_{ij}(t)\,Z_{ij}(t)dt \qquad (4.37)$$

De acordo com (4.25), tem-se:

$$U^{*}_{K+1,m}=U_I+U_{II}+\int_{t=t^{*}_{K+1,}}^{t=C(K+1,1)}Q(m,1)(t)dt \qquad (4.25)$$

com:

$$U_{II}=\sum_{i\in K}\sum_{j=1}^{M}\int_{t=C(K,1)}^{t=C(K,M)}Q_{ij}(t)Z_{ij}(t)dt \qquad (4.10)$$

A equação (4.10) pode ainda ser escrita na forma:

$$U_{II}=\sum_{i\in K}\sum_{j=1}^{M}\int_{t=C(K,1)}^{t=C(K+1,1)}Q_{ij}(t)\;Z_{ij}(t)\;dt+\sum_{i\in K}\sum_{j=1}^{M}\int_{t=C(K+1,1)}^{t=C(K,M)}Q_{ij}(t)\;Z_{ij}(t)\;dt$$

Chamando

$$U^{*}_{II}=\sum_{i\in K}\sum_{j=1}^{M}\int_{t=C(K+1,1)}^{t=C(K,M)}Q_{ij}(t)\;Z_{ij}(t)\;dt \qquad (4.38)$$

a equação anterior pode ser escrita como:

$$U_{II}=\sum_{i\in K}\sum_{j=1}^{M}\int_{t=C(K,1)}^{t=C(K+1,1)}Q_{ij}(t)\;Z_{ij}(t)\;dt+U^{*}_{II} \qquad (4.39)$$

Substituindo (4.39) em (4.25) e tendo em conta que o consumo de recurso é constante durante a execução das operações será:

$$U^{*}_{K+1,m}=U_I+\sum_{i\in K}\sum_{j=1}^{M}\int_{t=C(K,1)}^{t=C(K+1,1)}Q_{ij}(t)\;Z_{ij}(t)\;dt+U^{*}_{II}+Q(m,1)\;t(m,1) \qquad (4.40)$$

Substituindo (4.40) em (4.37), tendo em conta (4.38), será:

$$q\left(LB_{B,K+1,m} - C(K+1,1)\right) = D - U_I - \sum_{i \in K} \sum_{j=1}^{M} \int_{t=C(K,1)}^{t=C(K+1,1)} Q_{ij}(t)\, Z_{ij}(t)\, dt - Q(m,1)\, t(m,1)$$

Rearranjando a equação acima será:

$$LB_{K+1,m} = C(K+1,1) + \frac{D - U_I - \sum_{i \in K} \sum_{j=1}^{M} \int_{t=C(K,1)}^{t=t^{*}_{K+1}} Q_{ij}\, t_{ij}\, Z_{ij}(t) - Q_{K+1,1}\, t_{K+1,1}}{q} \qquad (4.41)$$

Na Figura 4.8 é apresentada a árvore de busca e os custos associados aos ramos associados a nós sondados. A seqüência de nós sondados está apresentada na Tabela 4.4.

4.4.4.3. Algoritmo C_1

O Algoritmo C_1 é uma fusão dos Algoritmos E e A, já que no algoritmo C_1 o valor do limitante inferior é definido ou através da estratégia de transformar a demanda futura de recurso compartilhado em tempo, ou através da heurística FMBB. Portanto, no exemplo em questão, serão observadas diferenças naqueles nós em que o limitante inferior calculado através da heurística FMBB superar o limitante inferior calculado através da estratégia proposta.

A árvore de nós sondados está apresentada na Figura 4.9.

4.4.4.4. - Algoritmo C_2

Para o exemplo em questão, o algoritmo C_2 utiliza a equação (4.41) para realizar a estimativa do limitante inferior considerando apenas o compartilhamento de um recurso. O limitante calculado usando a heurística FMBB é calculado a partir do instante $t^{*}_{K+1,m}$.

A aplicação do algoritmo C_2 aos dados do exemplo em questão resulta na árvore de busca apresentada na Figura 4.10.

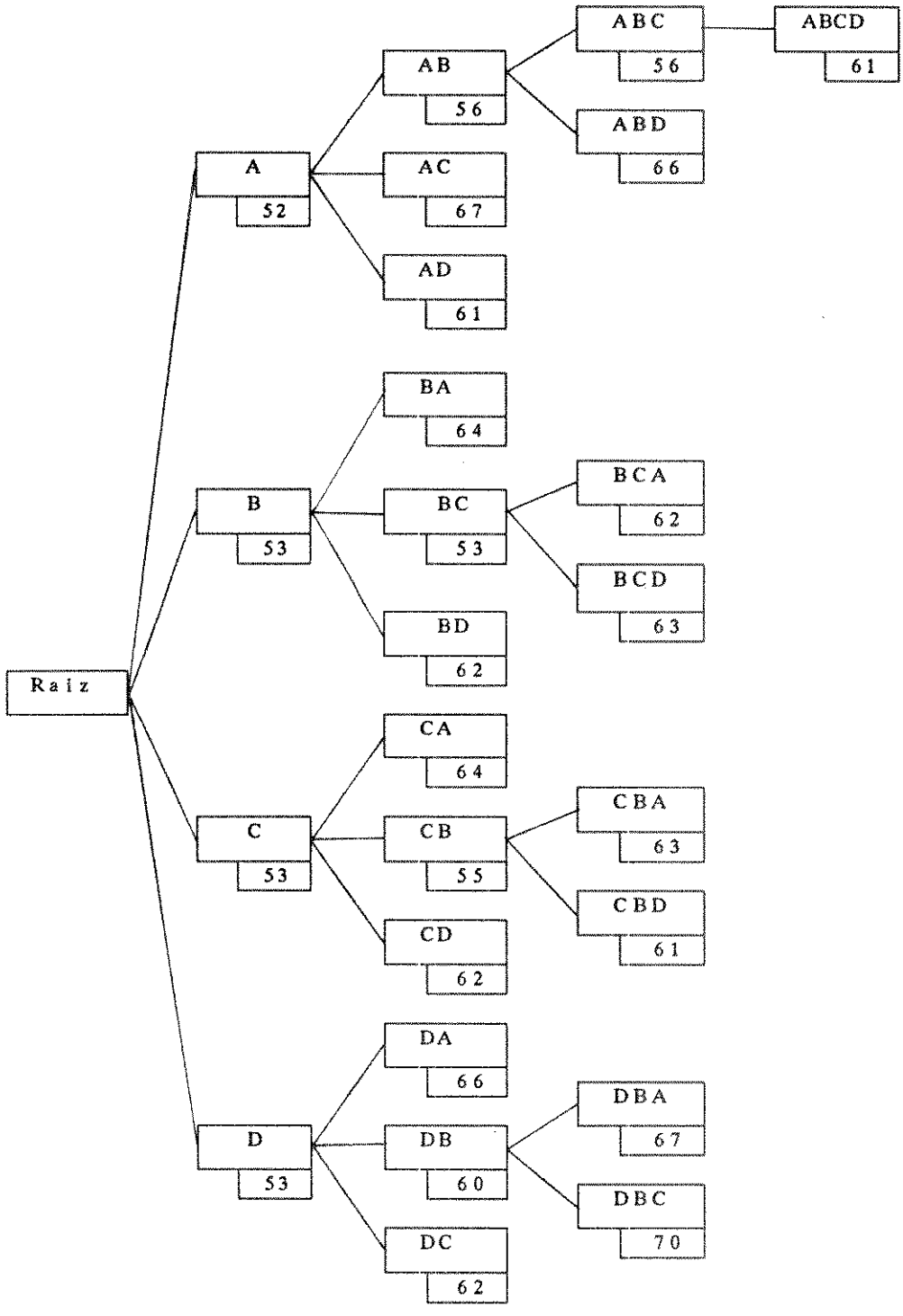


Figura 4.8 - Árvore nós sondados Algoritmo B

Tabela 4.4 - Seqüência de nós sondados (Algoritmo B)

Sequencia Parcial (K)	Tarefa Candidata (K+1)	Limit. Inferior $LB_{B,K+1,m}$	Limit. Inferior LB_B
1)A	B	55	
	C	60	
	D	52	52
2)B	A	64	
	C	53	53
	D	54	
3)C	A	63	
	B	53	53
	D	54	
4)D	A	61	
	B	53	53
	C	61	
5)AB	C	56	56
	D	57	
6)AC	B	67	67
	D	73	
7)AD	B	61	
	C	61	61
8)BA	C	71	
	D	64	64
9)BC	A	61	
	D	53	53
10)BD	A	62	62
	C	63	
11)BCA	D	62	62
12)BCD	-	63	63
13)CA	B	67	
	D	64	64
14)CB	A	60	
	D	55	55
15)CD	A	62	62
	B	62	62
16)DA	B	66	66
	C	70	
17)DB	A	60	60
	D	61	
18)DC	A	73	
	B	62	62
19)CBA	A	60	63
20)CBD	D	61	61
21)ABC	B	56	56
22)ABD	C	66	66
23)ABCD	-	61	61***
24)DBA	C	67	67
25)DBC	A	70	70

*** - Indica a solução ótima do problema

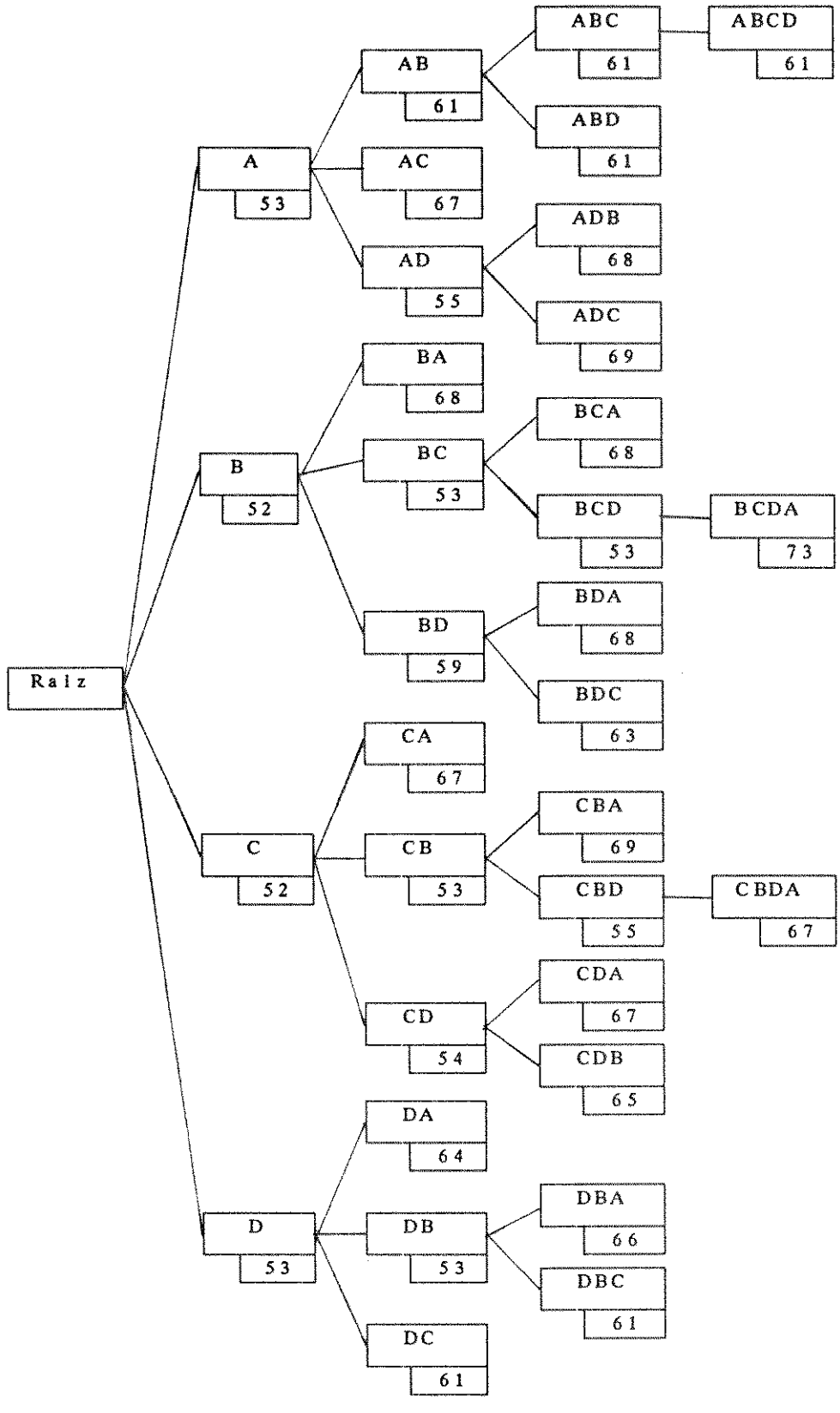


Figura 4.9 - Árvore de nós sondados - Algoritmo C_1

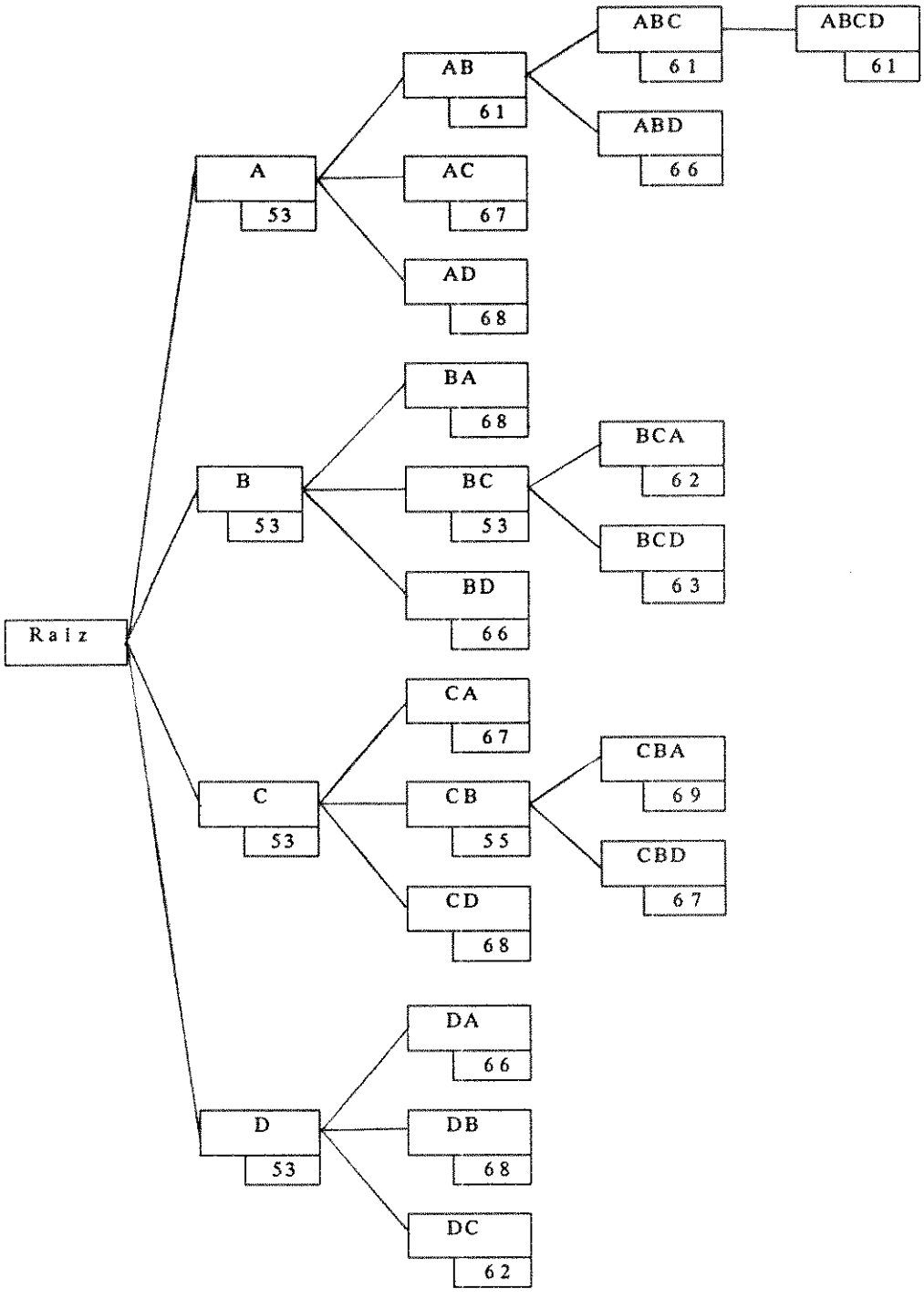


Figura 4.10 - Árvore nós sondados - Algoritmo C_2

4.5. Discussão dos Resultados

Na Tabela 4.6 são apresentados os valores do makespan de todas as seqüências do exemplo em questão em dois casos:

1. A oferta de recurso é ilimitada.
2. A oferta de recurso é constante limitada e igual a 10;

Tabela 4.6- Valores de makespan para cada seqüência possível

Sequencia	Makespan (oferta ilimitada)	Makespan (q=10)
ABCD	55	61
ABDC	55	75
ACDB	55	80
ACBD	55	67
ADBC	55	68
ADCB	55	69
BACD	54	80
BADC	54	82
BCDA	49	73
BCAD	49	68
BDAC	49	80
BDCA	49	80
CABD	50	75
CADB	50	82
CBAD	50	69
CBDA	50	67
CDAB	50	75
CDBA	50	81
DABC	47	72
DACB	47	78
DBAC	47	78
DBCA	47	78
DCAB	47	86
DCBA	47	78

A geração destas seqüências foi feita da seguinte maneira:

Caso 1. Quando a oferta de recurso é ilimitada, os instantes de término das tarefas obedece apenas a seguinte relação:

$$C(K,j)=\max\{ C(K-1,j),C(K,j+1)\} + t_{ij}$$

isto é, o instante factível para início da execução da $K^{\$sima}$ tarefa da seqüência no processador j, depende apenas do instante em que a tarefa K é terminada no processador (j-1) e da liberação do processador j pela tarefa (K-1).

Caso 2. Se a oferta de recursos é limitada, o instante de início da tarefa K no processador j é definida por uma relação do tipo (4.14). Esta relação significa que o instante de início de cada operação está vinculado não só aos instantes de liberação dos processadores, mas também do instante em que o perfil de demanda pode acomodar uma nova demanda.

Da Tabela 4.6 observa-se que os aumentos no valor do makespan entre os casos 1 e 2 não obedecem um padrão fácil de ser identificado.Por isso, um algoritmo do tipo D não oferece qualquer garantia de que a solução factibilizada estará nas proximidades da solução de partida, embora a solução de partida seja uma solução ótima para um flow shop sem limitação nos recursos. Desta Tabela observa-se igualmente que outras soluções poderiam ser utilizadas como solução de partida pois apresentam valores de makespan para o caso 1 iguais. No entanto os valores do makespan na presença de limitação de recursos são bastante diferentes. Em resumo, uma abordagem do tipo apresentada no Algoritmo D deve ser descartada.

A seguir procede-se a uma análise comparativa dos resultados obtidos com os algoritmos A, B, C_1, C_2 e E. Resumindo, estes algoritmos estimam a parcela do limitante inferior referente às tarefas não programadas a partir de:

A - excedente de recurso na região II e disponibilidade na região III;
B - excedente de recurso na região II utlizável já deduzido o valor mínimo de consumo por parte da primeira operação das tarefas candidatas a serem acrescentadas, e disponibilidade na região III;
C_1 - máximo entre os limitantes inferiores calculados pelos algoritmos A e E
C_2 - máximo entre os limitantes inferiores calculados com base no algoritmo B e o valor obtido da heurística FMBB;
E - heurística FMBB.

A parcela do limitante inferior referente às tarefas não programadas é obtida de forma idêntica em todos os algoritmos (o algoritmo E inclui em cada nó a compatibilização de recursos com as tarefas já programadas).

Para qualquer nó sondado em todos os algoritmos tem-se que:
- $LB_A \leq LB_{C_1}$ e $LB_B \leq LB_{C_2}$
- $LB_A \leq LB_B$
- $LB_{C_1} \leq LB_{C_2}$
- $LB_E \leq LB_{C_1} \leq LB_{C_2}$

No caso geral não podem ser definidas relações a priori entre:
- LB_B e LB_{C_1}
- LB_E e LB_A
- LB_E e LB_B

Nas Tabelas 4.7 e 4.8 são apresentados os valores de LB e os números de nós sondados em cada um dos algoritmos (A,B,C_1,C_2 e E).

Tabela 4.7 - Número de nós Sondados

Algoritmo	A	B	C_1	C_2	E
Número de Nós Sondados	38	25	33	23	33

Tabela 4.3 - Valores de LB para os algoritmos E, A, B, C_1 e C_2

Sequencia Parcial	E	A	C_1	B	C_2
1)A	53	51	53	52	53
2)B	49	52	52	53	53
3)C	50	52	52	53	53
4)D	47	53	53	53	53
5)AB	61	55	61	56	61
6)AC	67	60	67	67	67
7)AD	55	52	55	61	68
8)BA	68	64	68	64	68
9)BC	49	53	53	53	53
10)BD	59	54	59	62	66
11)CA	67	63	67	64	67
12)CB	50	53	53	55	55
13)CD	50	54	54	62	68
14)DA	64	61	64	66	66
15)DB	47	53	53	60	68
16)DC	61	61	61	62	62
17)ABC	61	56	61	56	61
18)ABD	61	57	61	66	66
19)ACB	-	60	-	-	-
20)ACD	-	62	-	-	-
21)ADB	68	64	68	-	-
22)ADC	69	61	69	-	-
23)BAC	-	-	-	-	-
24)BAD	-	-	-	-	-
25)BCA	68	61	68	62	68
26)BCD	49	53	53	63	63
27)BDA	68	62	68	-	-
28)BDC	63	63	63	-	-
29)CAB	-	-	-	-	-
30)CAD	-	-	-	-	-
31)CBA	69	62	69	63	69
32)CBD	50	55	55	61	67
33)CDA	67	62	67	-	-
34)CDB	65	62	65	-	-
35)DAB	-	-	-	-	-
36)DAC	-	-	-	-	-
37)DBA	66	60	66	67	-
38)DBC	61	61	61	70	-
39)DCA	-	-	-	-	-
40)DCB	-	-	-	-	-

Importância da Região II:

A importância da análise de conflitos na região II pode ser mostrada através da comparação dos resultados obtidos pelos algoritmos C_1, C_2 e E e A e B.

C_1 versus E e C_2 versus E

Tem-se $LB_{C_1} > LB_E$ em apenas 9 nós (de um total de 33). Isto significa que o aumento de LB no algoritmo C, em relação ao E, originado pela previsão de conflitos pelos recursos entre as tarefas não programadas feita pelo algoritmo C_1, tem importância relativamente pequena (em termos de quantidade de seqüências parciais para as quais este aumento existe). Isto faz com que o número de nós sondados seja semelhante.

Tem-se $LB_{C_2} > LB_E$ em 14 nós (de um total de 23 nós comuns a C_2 e E). Este número de casos grande significa que a previsão de conflito feita pelo algoritmo C_2 é relevante em contraposição com a previsão feita pelo algoritmo C_1. O número de nós sondados diminui sensivelmente (33 versus 23).

A versus B

A segunda observação pode ser também verificada comparando os valores de LB para os algoritmos Ae B, neste caso é claro sem o recurso à comparação com o algoritmo E, poeque os algoritmos A e B não levam em conta os tempos de processamento na estimativa de LB. Tem-se $LB_B > LB_A$ em 20 nós (de um total de 25 nós comuns a A e B). O número do nós sondados diminui também (38 versus 25), mostrando que a previsão de conflitos realizada pelo algoritmo B é importante.

A ineficiência das abordagens propostas nos Algoritmos A e C_1 ao problema em questão é derivada da utilização de uma heurística incompleta para estimar o makespan em cada nó. A estimativa do tempo necessário para a execução das tarefas não programadas é feita com base exclusivamente no volume de recursos ainda necessários para a execução destas tarefas. No entanto a restrição (3.1) impõe que o perfil deve ser capaz de acomodar também a demanda instantânea de recurso. Esta restrição é utilizada para compatibilizar a demanda das tarefas já programadas, mas não para realizar estimativas de makespan.

A proposta contida nos Algoritmos B e C_2 é uma proposta nesta direção, na medida que procura verificar a possibilidade do perfil consolidado acomodar pelo menos a primeira operação da tarefa candidata.

A estratégia proposta reforça a idéia apresentada no Capítulo 3 que a natureza dinâmica do scheduling em flow shops aproxima este problema ao problema do scheduling em job shops quanto à natureza das propostas de solução a serem empregadas. A fusão entre os níveis de seqüenciamento e alocacação de recursos faz com que certamente a dimensão do problema de scheduling em flow shops não seja apenas N!, mesmo nos caso em que sejam mantidas exclusivamente as seqüências de permutação. Será então necessário realizar operações sobre os perfis de demanda e oferta, para auxiliar na decisão de qual caminho adotar na busca de soluções ótimas ou implementáveis. Estas operações realizadas sobre os perfis de oferta e demanda revelam que os recursos compartilhados aumentam a dimensão do problema e sua complexidade.

Importância relativa dos recursos e do encadeamento das operações

A comparação entre os resultados obtidos com os algoritmos A e C_1,

e B e C_2, permite analisar a conseqüência da introdução do encadeamento das operações no cálculo do limitante inferior (através da heurística FMBB).

A partir das Tabelas 4.7 e 4.8 verifica-se que $LB_{C_1} > LB_A$ em 18 nós (de um total de 31) e $LB_{C_2} > LB_B$ em 12 nós (de um total de 23). O número de nós sondados tem uma diminuição entre A e C_1 (38 versus 33) e entre B e C_2 (25 versus 23).

Existe nos dois casos um aumento de eficiência, mas relativamente a melhora de C_1 em relação a A é maior que a de C_2 em relação a B, como pode ser visto pelo número de nós sondados. Este fato sugere que, devido à oferta limitada de recursos, a estimativa mais correta do limitante inferior feita pelo algoritmo B (e C_2) foi suficiente, neste exemplo, para acomodar as conseqüências do encadeamento de tarefas, característica esta acrescentada pelo algoritmo C_2.

Algoritmo A versus Algoritmo E

Uma comparação entre as árvores de busca relativas aos algoritmos A e E apresentadas nas Figuras 4.8 e 4.7 respectivamente, mostra que o número de nós visitados pelo algoritmo A é maior do que o número de nós visitados pelo algoritmo E. De fato este comportamento pode ser explicado pois, à medida que se aproxima o final da árvore, a competição pelos recursos comuns tende a ser reduzida. Entre os níveis (N-1) e N da árvore de busca, o efeito sobre o makespan do tempo necessário para executar a última tarefa da seqüência, deve se sobrepor ao tempo calculado com base na oferta de recursos comuns.

Concluindo, os algoritmos B e C_2 tiveram melhor desempenho na situação exemplificada de oferta crítica de recursos (Figura 4.4) mostrando que a correção do limitante inferior através do instante factível de início da primeira operação e do consumo de recurso por esta operação é importante. Sendo que o aumento de esforço computacional entre C_2 e B é pequeno, propõe-se a utilização do algoritmo C_2 que acrescenta impactos do encadeamento de tarefas ao cálculo do limitante inferior.

Em situações menos críticas sobre a oferta de recursos ou com maior número de tarefas e demanda individual de recursos menor, o desempenho dos algoritmos A e C_1 deve ser mais próximo do desempenho de B e C_2, com um esforço computacional menor.

A abordagem proposta nos Algoritmos A, B, C_1 e C_2 certamente não são adequadas para a solução de todos os problemas de scheduling em flow shops, mas permitem evidenciar algumas das principais dificuldades encontradas na solução do problema, e constituem um passo a mais na busca de soluções eficientes para o problema de scheduling em flow shops.

CAPÍTULO 5 - CONCLUSÕES

Neste trabalho o objetivo foi apresentar um panorama do scheduling em flow shops, seus principais problemas, e desenvolver uma abordagem para a geração de schedules ótimos quando há limitação na oferta de recursos compartilhados, excetuando-se os processadores.

A abordagem proposta não esgota certamente as alternativas de solução do problema, ao contrário, põe em maior evidência as dificuldades associadas à busca de soluções factíveis, já que a abordagem de programação matemática parece estar, ao menos temporariamente, descartada.

A principal diferenciação apresentada entre os problemas de scheduling em flow shops e job shops é com relação à "dimensão". O problema de scheduling em flow shops é normalmente apresentado como um problema de dimensão N!, quando são sondadas apenas as seqüências de permutação. O problema de scheduling em job shops, dada a característica do fluxo interno de tarefas, tem certamente uma dimensão $N!^M$. Esta diferença é certamente verdadeira quando se trata de problemas de seqüenciamento em flow shops. Quando se trata de um problema de scheduling em flow shops, a abordagem proposta evidencia, através da relação (4.14), a necessidade de uma política de alocação de recursos que põe em evidência aspecto temporal do sequenciamento na presença de limitação na oferta de recursos compartilhados.

No caso do seqüenciamento este aspecto temporal não é tão evidente, embora exista, pois está normalmente associado à política de armazenagem intermediária. Visto que o problema de flow shops normalmente reportado na literatura é o flow shop UIS, o problema de alocação parece ser secundário.

No caso de compartilhamento de recursos a importância do número de processadores (M) passa a ser maior não figurando exclusivamente cálculo da função de custo adotada, mas também nas decisões a serem tomadas em cada nó da árvore.A influência dos instantes de início sobre o valor do makespan evidencia a importância desta dimensão no problema no problema de scheduling.

A abordagem proposta no Capítulo 4 mostrou a importância que possa vir a ter uma abordagem desenvolvida para o problema de scheduling. É possível, como se viu, resolvê-lo recorrendo a outras estratégias tais como os Algoritmos D e E, que podem conduzir ou a soluções inadequadas, como é o caso do algoritmo D, ou estratégias que podem exigir muito tempo para conduzir a uma solução ótima ou implementável.

A abordagem proposta (Algoritmo B ou C_2) parece ser adequada para a solução do problema, embora precise ter seu desempenho testado com vários tipos de dados de processamento, especialmente com relação ao número de processadores e tarefas.

O incremento em eficiência da abordagem proposta parece ser resultante de um conhecimento mais detalhado da Região II.

Para situações em que a Região II seja muito extensa, o aumento no valor do limitante resultante da identificação do instante mais cedo para início da primeira operação da tarefa candidata pode ser muito pequeno.

Uma das limitações da abordagem proposta é certamente o cálculo do limitante inferior baseado apenas no instante de início da primeira operação da tarefa candidata. Dependnedo do comportamento do perfil de demanda na Região II, outras operações da mesma tarefa podem gerar atrasos significativos. Seria portanto desejável uma abordagem capaz de fornecer informações mais amplas acerca de prováveis conflitos na Região II e seus reflexos na estimativa do makespan.

Uma abordagem que parece ser adequada para ser usada em sistemas de scheduling é o Ordenamento de tarefas orientado por recursos apresentado no Capítulo 3. Um passo inicial neste sentido foi dado com a proposta de adaptação desta abordagem para o seqüenciamento de tarefas em flow shops detalhada no Anexo C.

Durante a apresentação deste trabalho foi discutida a importância da armazenagem intermediária no processamento serial multiestágios. Embora muitos trabalhos tenham sido desenvolvidos com o objetivo de calcular "completion time" na presença de armazenagem limitada, não há praticamente propostas de alocação de armazenagem intermediária. Infelizmente a abordagem proposta não é adequada para sua aplicação a problemas onde o recurso compartilhado seja a armazenagem. Ela têm um comportamento especial difícil de ser tratado através das abordagens normalmente apresentadas na literatura. Este seu comportamento especial vem do fato que não existe um perfil de demanda deste recurso, muito embora haja um perfil de oferta. A demanda ïé dinamicamente criada em função da limitação dos demais recursos, especialmente os processadores. Uma abordagem do tipo apresentada no Anexo C pode ser útil no sentido de criar perfis de demanda potencial deste recurso, de forma a permitir a orientação da escolha do caminho mais adequado otimizando a função de custo adotada.ï

Do conjunto de observaçãese resultados obtidos neste trabalho, podem ser destacados os seguintes tópicos para o desenvolvimento de trabalhos futuros:

1)Aplicação da abordagem proposta a vários tipos de problemas de scheduling em flow shops com o objetivo de inferir a importância da identificação do instante mais cedo de início da tarefa candidata ($t_{K+1,m}$) em função da fragmentação do perfil de demnada, isto é, qual o desempenho relativo dos algoritmos C_1 e C_2 em função do número de operações de cada tarefa e das demandas individuais das operações serem muito diferentes entre si.

2) Pesquisar abordagens que permitam fazer algum tipo de avaliação de conflitos na Região III com o objetivo de melhorar a estimativa do limitante inferior das abordagens propostas.

3) Adquisição de conhecimentos sobre o problema para o desenvolvimento de procedimentos heurísticos que possam abreviar a busca na árvore.

Anexo A - Determinação do instante de término das tarefas (Completion Time)

Este é um problema cuja solução é indispensável para o cálculo da função objetivo, o qual é necessário resolver repetidas vezes qualquer que seja a abordagem empregada(Wiede(1),(2),(3),Ku e Karimi(1986), Rajagopalan(1987)

No caso de serem empregados métodos de busca, a cada nó da árvore corresponde uma seqüência parcial, e consequentemente um perfil temporal de utilização dos recursos disponíveis (processadores, armazenagem, utilidades etc). No caso de ser empregada uma abordagem de programação matemática do problema, será necessário desenvolver um modelo matemático capaz de manipular as restrições de alocação tendo em conta os conflitos temporais na demanda de recursos comuns. É evidente então a necessidade de calcular também nesta abordagem os "completion time".

Normalmente são utilizadas expressões analíticas e/ou relações de recorrência para o cálculo do instante de término das tarefas. O problema de determinação do completion time para processos seriais pode ser definido como:

"Dada a seqüência de produtos em todas as unidades e dados:

1.tempos de processamento t_{ij};

2.tempos de transferenciaa $_{ij}$;

3.tempos de preparação s_{ijk};

4.número de unidades de armazenamento;

4.Produção em batelada;

5.Política de armazenagem;

6.Política de gerenciamento de conflitos na demanda de recursos compartilhados;

derivar relações de recorrência para C_{ij} , onde:

C_{ij}= instante no qual o produto i começa a ser transferido da unidade j.

No caso geral os tempos de transferência entre equipamentos adjacentes, de equipamentos para a armazenagem e da armazenagem para os equipamentos não são nulos. A agregação dos diferentes tempos especificados nas receitas dos produtos depende de diversos fatores tais como importância relativa entre os tempos de processamento e transferência e distorção do makespan gerado pela agregação destes tempos.

Quando a armazenagem intermediária for ilimitada e os recursos comuns não sofrerem qualquer tipo de restrição o problema de cálculo do "completion time" é conceitualmente mais simples na medida que basta definir o instante de início da operação e imediatamente o instante final é também conhecido. O problema torna-se formalmente e conceitualmente mais complexo quando a armazenagem e/ou recursos comuns forem limitados.

Neste caso, duas condições devem ser consideradas:

1) O fato de um processador estar livre não significa que ele possa ser imediatamente utilizado. Sua utilização dependerá também da capacidade do sistema de recursos absorver mais esta demanda.

2) O fato de uma operação estar terminada não significa que o

processador será imediatamente liberado para a excução de outra operação. Sua utilização dependerá da liberação do estágio seguinte ou da armazenagem intermediária (se houver).

Estas duas condições mostram que, em um problema de scheduling, a ocupação do processador não será ditada exclusivamente pelo tempo de processamento, mas sim por um tempo de residência da operação no processador.

O cálculo dos "completion time" que serão apresentadas a seguir foram desenvolvidas para aqueles casos em que não há restrições especiais, tais como restrições de estabilidade, ou condições de processamento mistas, tais como blocos de processadores sendo operados segundo diferentes políticas de processamento.

A.1. Armazenagem Intermediária

Existem duas situações importantes que devem ser analisadas: armazenagem inter-estágios e armazenagem compartilhada.

A) Armazenagem inter-estágios

Para se determinar o instante de término da tarefa i no processador j devem ser consideradas quatro condições possíveis:

1º) O produto i deve estar terminado na unidade (j-1) então:

$$C_{ij} \geq C_{i(j-1)} + a_{i(j-1)} + t_{ij} = \psi_1 \qquad (A1.1)$$

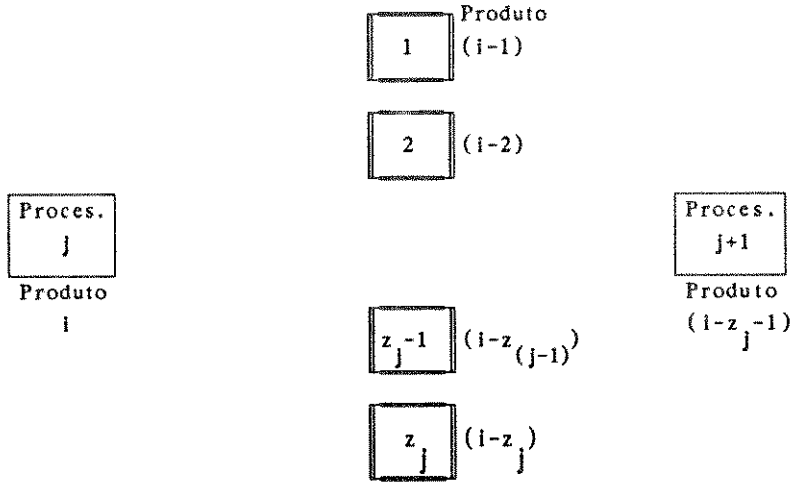
2º) O produto (i-1) deve estar terminado na unidade j

$$C_{ij} \geq C_{(i-1)j} + a_{(i-1)j} + s_{(i-1)ji} + a_{i(j-1)} + t_{ij} = \psi_2 \qquad (A1.2)$$

3º) Não existe armazenagem (NIS), de tal maneira que o produto deve esperar que a unidade (j+1) seja liberada:

$$C_{ij} \geq C_{(i-1)(j+1)} + a_{(i-1)(j+1)} + s_{(i-1)(j+1)(i-2)} = \psi_3 \qquad (A1.3)$$

4º) Não existe armazenagem livre, e o produto i deve esperar que uma unidade de armazenagem esteja livre. Esta situação está apresentada na ilustração a seguir.



$$C_{ij} \geq C_{(i-z_j-1)(j+1)} + a_{(i-z_j-1)(j+1)} + s_{(i-z_j-1)(j+1)(1-z_j)} + a_{(i-z_j)j} = \psi_4 \quad (A1.4)$$

Então será:

$$C_{ij} = \max \{\psi_1, \psi_2, \psi_3, \psi_4\} \quad (A1.5)$$

A condição 4 é aquela em que deve ser desenvolvido simultaneamente a programação das tarefas e a programação da armazenagem.

Uma situação não prevista nestas quatro condições é aquela em que algum produto sofra restrição de estabilidade de forma que logo que o produto esteja pronto, deve ser transferido para o processador seguinte. Neste caso tem-se um sistema com política de armazenagem mista (FIS/NIS) e a estratégia mais frequentemente empregada é agregar os processadores em blocos de tal forma que no final de cada bloco os produtos intermediários sejam sempre estáveis (Karimi (1989))

B)Armazenagem compartilhada entre estágios

Para o caso da armazenagem compartilhada entre estágios o problema é muito mais complexo. No caso anterior as estratégias utilizadas necessitavam apenas que fossem registrados os eventos de liberação dos equipamentos, isto é, mesmo no caso de existir armazenagem intermediária, para se calcular o instante de término da tarefa i no processador j não é necessário conhecer o perfil completo de utilização da armazenagem, mas apenas se existe tanque disponível naquele instante. Para o caso de um flow shop com armazenagem compartilhada submetido a uma política mista FIS /NIS as relações básicas de recorrencia são:

$$C'_{ij} = \max \left[C_{i(j-1)}, E_{(i-1)j} \right] + a_{i(j-1)} + t_{ij} \quad (A1.6)$$

$$C_{ij} = \max \left[C_{i(j-1)}, E_{(i-1)j} \right] + a_{i(j-1)} + r_{ij} \quad (A1.7)$$

onde:

C'_{ij} = instante em que a tarefa i está pronta para deixar o processador j

C_{ij} = instante em que a tarefa i começa ser transferida do processador j

E_{ij} = instante em que o processador j está pronto para receber a tarefa $(i+1)$

r_{ij} = tempo de residencia da tarefa i no processador j

então:

$$C_{ij} = C'_{ij} + (r_{ij} - t_{ij}) \quad (A1.8)$$

$$E_{ij} = C_{ij} + a_{ij} + s_{ij(i+1)} \quad (A1.9)$$

Como se vê, r_{ij}, tempo de residência da tarefa i no processador j, é o tempo de processamento somado ao tempo que a tarefa i deve esperar no processador j antes que ela comece a ser transferida para o processador (j+1).

O valor de r_{ij} dependerá da política adotada para a ocupação da armazenagem. Assim será:

- $r_{ij}=t_{ij}$ se o processador (j+1) está livre;
- Qual deverá ser o valor de r_{ij} se a unidade de armazenagem está livre e o processador (j+1) está ocupado, ou no caso em que a armazenagem e o processador estão simultaneamente ocupados.
- Se dois ou mais produtos demandam armazenagem ao mesmo tempo, qual a política a ser adotada.

Nestas situações são gerados conflitos de armazenagem que devem ser solucionados quer através de uma heurística de alocação ou outra abordagem. O principal problema será então utilizar uma abordagem capaz de calcular, e se for o caso, minimizar, r_{ij} (tempo de residência da tarefa i no processador j).

Para o caso de existir mais de um tanque de armazenagem intermediária, o problema será mais complexo envolvendo ainda a estratégia de definir quando, e qual produto e qual unidade de armazenamento deverá ser utilizada.

A.2. Recursos Compartilhados

O problema de calcular o "completion time" quando os recursos compartilhados são limitados se assemelha ao da armazenagem intermediária compartilhada entre estágios.A definição do instante em que a tarefa libera o processador ou a armazenagem intermediária depende da análise das seguintes condições:

1) A unidade (j+1) está livre para iniciar o processamento da tarefa i. Em caso positivo, deve ser verificado o instante em que o perfil de consumo do recurso pode acomodar o início do processamento.
2) Se a unidade (j+1) não estiver livre, verificar se existe tanque de armazenagem disponível para receber o produto i. Caso contrário, o processador permanece bloqueado até que haja armazenagem livre ou o processador possa iniciar o processamento.

Na abordagem apresentada no Capítulo 4, antes de iniciar o cálculo do makespan, é necessário conhecer o perfil consolidado de consumo de recursos da seqüência parcial no nó. Naquele caso não há limitação na armazenagem intermediária, de forma que o instante de início factível para início da operação era definido pelos seguintes instantes:

1) Instante em que a operação anterior da mesma tarefa estiver terminada;
2) Instante em que o processador requerido para executar a operação candidata estiver livre;
3) Instante mais cedo que o perfil de oferta puder acomodar a nova demanda.

Estas condições irão gerar uma relação do tipo (4.15) apresentada no Capítulo 4.

A limitação na oferta de recursos compartilhados gera interações complexas com a política de armazenagem intermediária. No caso em questão,

esta interação pode ser ignorada na medida que se admite a existência de tanques de armazenagem com volume suficiente para atender qualquer demanda. De fato, a limitação na oferta de recursos, se ela for importante, exigirá tempos de residência maiores na armazenagem intermediária enquanto se aguarda a possibilidade do perfil de oferta poder acolher uma nova demanda.

A.3. Comentários

A importância do cálculo do completion time é evidente em todos os problemas de seqüenciamento e scheduling em flow shops.

A complexidade do cálculo do completion time depende da natureza do problema de programação da produção. A conseqüência mais importante é o número elevado de equações necessários para exprimir, através de expressões matemáticas, as restrições de ocupação dos processadores. Esta dificuldade pode ser confirmada pela ausência de formulações matemáticas dos problemas de scheduling. As abordagens que recorrem a métodos de busca exigem apenas o conhecimento de fórmulas de recorrência e condições tais como as discutidas neste anexo que permitem decidir qual a decisão a ser tomada no instante de ocorrência do conflito.

Anexo B - Estratégias de Alocação

A utilização de algoritmos heurísticos para o scheduling de tarefas, tais como os apresentados no Capítulo 3 ou o algoritmo D apresentado no Capítulo 4, exigem que, na ocorrência de um conflito temporal de demanda de um recurso comum, seja adotada uma estratégia de solução do conflito na medida que duas ou mais operações demandam simultâneamente o mesmo recurso. A estratégia empregada para solucionar o conflito não é, normalmente, parte integrante do método e precisa ser escolhida pelo usuário.

A estratégia utilizada para solucionar o conflito influencia o valor final do critério de desempenho adotado, mas até o presente não há qualquer razão que justifique a superioridade de algum tipo de estratégia.

A seguir serão apresentadas em linhas gerais duas estratégias de alocação freqüentemente utilizadas na literatura:

1) Estratégia de prioridade do evento

-Fila do tipo "primeiro a solicitar o recurso primeiro a ser servido";

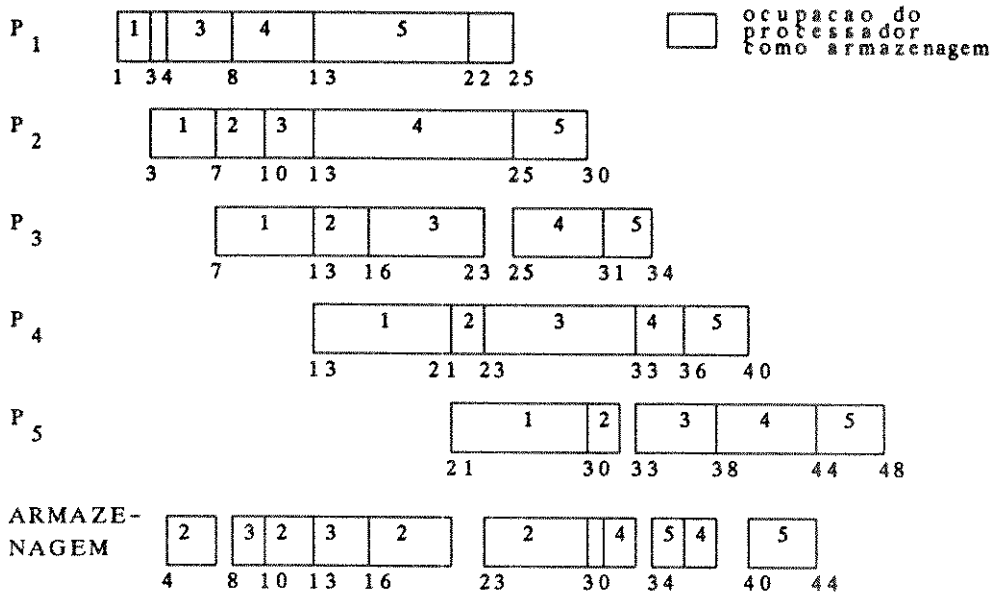
-Sempre que a demanda ocorre e o recurso está disponível, ocorre a alocação;

2) Estratégia de prioridade do Produto

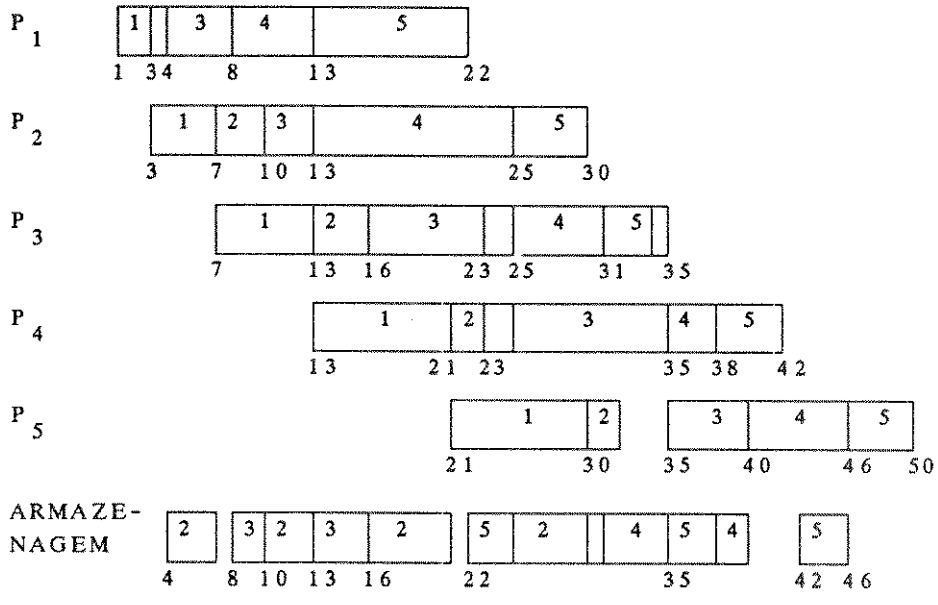
-Alocar o recurso de acordo com prioridade imposta pela seqüência, isto é, a ordem da tarefa na seqüência rege igualmente a ordem com que o recurso é utilizado.

Na Figura A.1 são apresentados duas programações de uma seqüência de ordenamento de tarefas obedecendo as duas estratégias citadas usadas para administrar conflitos na armazenagem intermediária, sendo mostrada a diferença no valor do makespan em cada caso.

Para o caso específico da armazenagem intermediária inter- estágios e no caso de existir mais de um tanque de armazenagem intermediária, o problema é mais complexo envolvendo ainda a estratégia de definir quando e qual produto e qual unidade de armazenamento deverá ser utilizada.



a) Schedule baseado na prioridade do produto



b) Schedule baseado na prioridade do evento

Figura A.1 - Estratégias de alocação

Uma alternativa é utilizar testes de disponibilidade da armazenagem para decidir a alocação do recurso. Dois procedimentos que podem ser empregados nesta situação são:

Procedimento 1

- Os perfis de demanda de cada unidade devem ser retidos;
- Na ocorrência da demanda de armazenagem cada unidade deve ser checada;
- Alocar a primeira unidade de armazenamento disponível.

Procedimento 2

- Manter perfis combinados de utilização de todas as unidades de armazenamento;
- Manter acessível o número de unidades disponíveis em cada instante;
- Não alocar uma unidade específica mas apenas indicar a necessidade de utilização do recurso em um intervalo de tempo.
- Decidir posteriormente qual será a unidade utilizada e por qual produto.

Os problemas apresentados brevemente neste anexo com respeito à dificuldade de utilizar uma estratégia de alocação da armazenagem intermediária, são semelhantes aos problemas de alocação de recursos compartilhados. Não há até o presente qualquer maneira de inferir qual o resultado no custo final da seqüência sem realizar a simulação completa dos eventos.

Anexo C - Exemplo de Aplicação do Scheduling Orientado por Recursos no Sequenciamento de Tarefas em Flow Shops

Neste Anexo será apresentada e discutida a aplicação da metodologia proposta por Sycara et al. para job shops ao sequenciamento de tarefas em um flow shop UIS, e a possibilidade de aplicação desta abordagem a outros problemas de scheduling em flow shops.

O objetivo é seqüenciar três tarefas, cada uma constituída de três operações, estando disponíveis três processadores, cada um indicado para executar uma das operações. O fluxo de tarefas é unidirecional caracterizando um flow shop. Os tempos de processamento estão apresentados na Tabela B.1.1.

Tabela B.1.1 - Tempos de processamento

Tarefa	Processador		
	1	2	3
1	8	7	5
2	4	9	6
3	5	3	7

A abordagem de alocação proposta é composta de duas partes: A) Ordenamento de Operações e B) Ordenamento de Alocações que são aplicadas ciclicamente sendo que, a cada ciclo, deve ser garantida a factibilidade das alocações através da atividade de propagação das restrições.

B1) Ordenamento das Operações

Na abordagem proposta, as operações são ordenadas segundo medidas de "criticalidade" em termos de demanda do recurso compartilhado, quer dizer, operações mais críticas tem prioridade na alocação do recurso. No caso em questão, o recurso compartilhado é o próprio processador, disponível na forma de um recurso unitário.

A construção das janelas de tempo iniciais factíveis para a execução das operações.através da abordagem proposta exige que sejam definidas:

- Restrições de precedência, que fixam os instantes possíveis de inicialização das operações;
- Prazos de entrega de cada tarefa ou horizonte de tempo em que se deseja desenvolver o scheduling;

Para analisar o desempenho desta abordagem foi fixado como prazo de entrega o instante t=30 correspondente ao makespan da seqüência de permutação ótima em um flow shop UIS, pois é impossível o processamento das tarefas em um prazo inferior a este.

As fórmulas de recorrência que permitem a determinação dos instantes de abertura e fechamento das janelas iniciais para o flow shop são:

$$DF_{ij} = h_i - \sum_{l=j+1}^{M} t_{il} \qquad (B.1.1)$$

$$DI_{ij} = R_i + \sum_{l=1}^{j-1} t_{il} \qquad (B.1.2)$$

onde:

DI_{ij}= instante de abertura da janela inicial da tarefa i no processador j

DF_{ij}= instante de fechamento da janela inicial da tarefa i no processador j

h_i= instante máximo em que a tarefa i deve estar terminada

R_i= instante mínimo em que a tarefa i pode ter seu processamento iniciado

t_{il}= tempo de processamento da tarefa i no processador l

M= número de operações

A duração da janela (DW_{ij}) é dada então por:

$$DW_{ij} = DF_{ij} - DI_{ij}$$

No caso em questão será:

$h_1=h_2=h_3=M=30$

$R_1=R_2=R_3=0$

Na Figura B.1.1 são apresentadas as janelas temporais tecnológicamente factíveis das operações de cada tarefa.

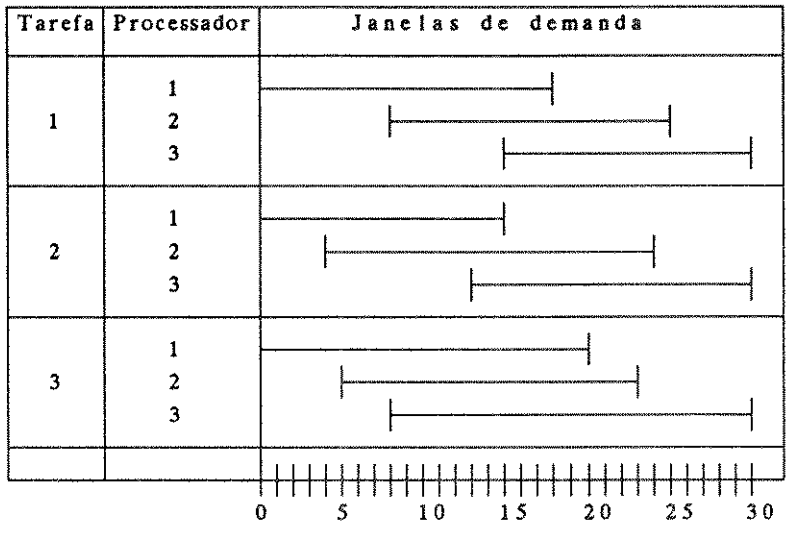


Figura B.1.1 - Janelas iniciais de cada tarefa em cada processador

O passo seguinte consiste em construir as curvas de criticalidade de cada operação. A construção destas curvas foi feita em intervalos de tempo iguais a um.

A construção destas curvas é baseada em duas quantidades:

1) No número de alocações que podem ser construídas para acomodar a

execução da operação na janela de demanda: np_{ij};

2) No número de vezes que cada intervalo de tempo, k, tem chances de ser utilizado para a alocação da operação na janela de demanda: nt^{k}_{ij}.

Na Figura B.1.2 é apresentada a janela inicial da operação 2 da tarefa 2, juntamente com as informações necessárias para a construção da curva de demanda probabilística.

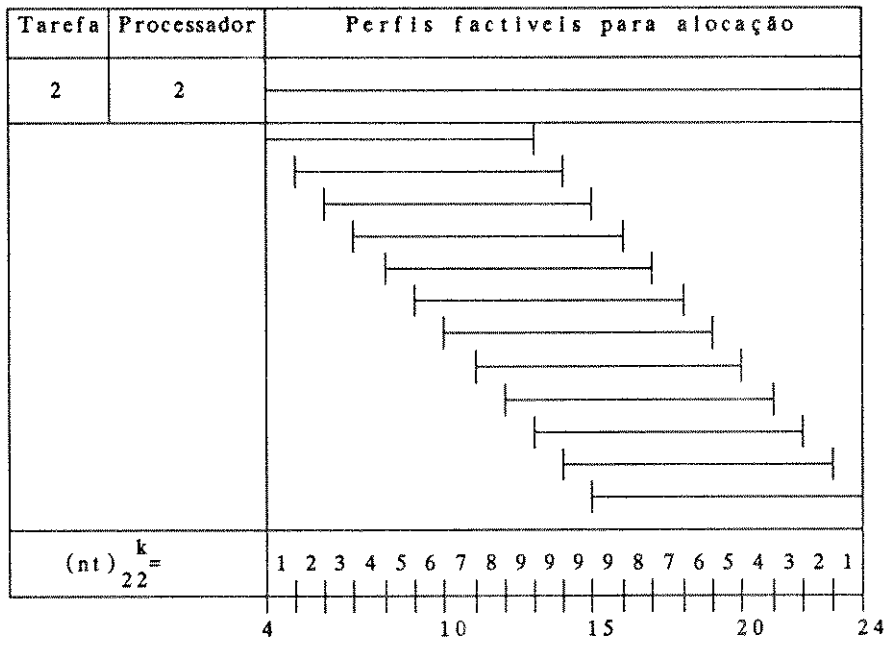


Figura B.1.2 - Construção dos perfis factíveis de alocacão na janela de demanda

Os valores de np_{ij} são calculados através da expressão:

$$np_{ij} = (DF^{l}_{ij} - DI^{l}_{ij}) - t_{ij} + 1 \qquad (B.1.3)$$

de forma que a probabilidade associada a cada intervalo de tempo será:

$$p^{k}_{ij} = \frac{nt^{k}_{ij}}{np_{ij}} \qquad (B.1.4)$$

O valor máximo de p^{k}_{ij} é denominado criticalidade da operação.

Na Figura B.1.3 é apresentada a curva de criticalidade da operação 2 da tarefa 2.

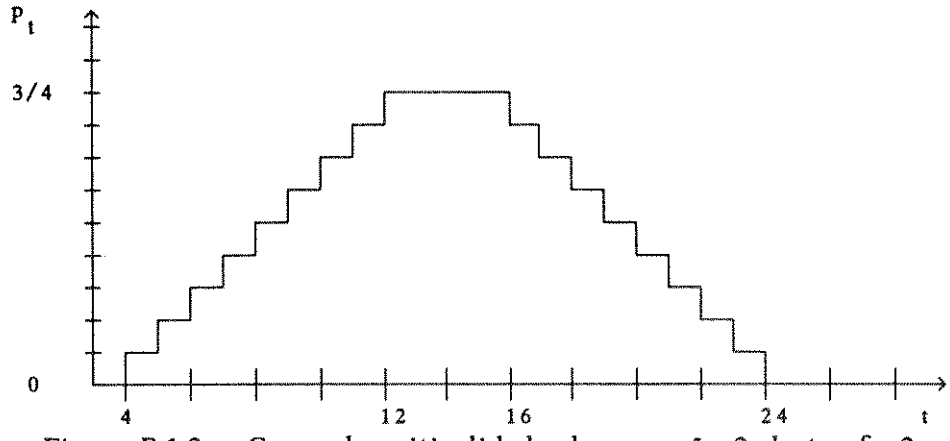


Figura B.1.3 - Curva de criticalidade da operação 2 da tarefa 2

O passo seguinte consiste construir as curvas de demanda agregada de cada processador para identificar o "pico de contenção", isto é, identificar o instante em que ocorre a maior demanda global e qual o processador que apresenta esta demanda. As curvas de demanda agregada são obtidas através da soma das criticalidades de todas as operações que demandam um processador específico. Portanto são curvas construídas para cada processador. Na Figura B.1.4 é apresentado um exemplo de curva de demanda agregada e o respectivo pico de contenção.

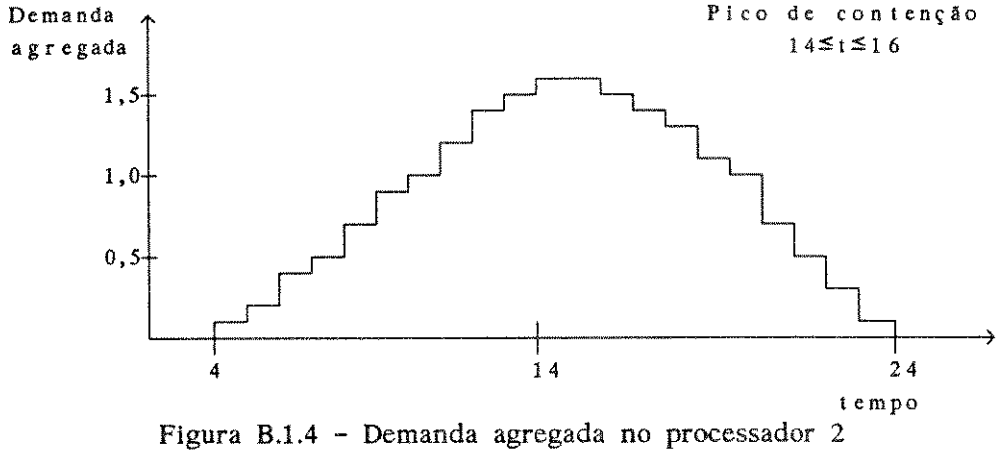


Figura B.1.4 - Demanda agregada no processador 2

Após a determinação do pico de contenção, procede-se à identificação, dentre as diversas operações que demandam este processador no intervalo de tempo em que ocorre a demanda máxima agregada, daquela que apresenta a maior criticalidade. A operação assim identificada é a escolhida como a próxima a ser alocada.

No exemplo em questão, o processador com maior demanda acumulada é o processador 2, e o intervalo de tempo em que ocorre o pico de contenção é [14 - 16]. Neste intervalo de tempo, a operação mais crítica é a operação 2 da tarefa 2.

A próxima etapa corresponde ao ordenamento das alocações, isto é, uma vez decidida qual a próxima operação a ser alocada, é necessário decidir em qual instante de tempo a operação deverá ser iniciada.

B.2)Ordenamento das Alocações

Sycara et al. propõem duas heurísticas para o ordenamento das alocações. Neste exemplo foi utilizada a heurística LCV - Least-Constraining Value - que consiste em alocar a operação mais crítica de forma a criar o menor impacto sobre o perfil de demanda.

A heurística LCV consiste escolher o instante correspondente à probabilidade de menor impacto para o início da alocação da operação. Atribui-se uma medida de impacto a cada instante possível para a inicialização da operação mais crítica na janela de demanda. Este impacto é estimado considerando o número de operações que demandam este intervalo de tempo e a demanda global de intervalos de tempo por estas operações. Quanto menor o impacto associado a um intervalo de tempo, maior a probabilidade deste intervalo ser escolhido para o início da operação mais crítica.Na Figura B.1.5 são apresentados os valores desta probabilidade associada a cada instante de tempo da janela de demanda sendo igualmente indicado o instante escolhido para o início da operação.

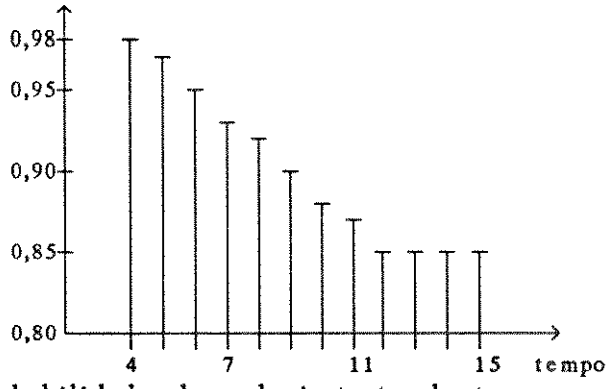


Figura B.1.5 - Probabilidade de cada instante de tempo ser escolhido para o início da operação mais crítica

De acordo com a heurística adotada, a operação 2 da tarefa 2 deve ser iniciada em t=4. Esta alocação cria uma restrição temporal de bloqueio do processador no intervalo [4,13] que impede sua utilização por qualquer outra operação neste intervalo. Esta restrição deve ser propagada para as demais operações de forma a preservar a factibilidade tecnológica do flow shop.

B.3) Propagação de Restrições

A alocação faz com que a janela de tempo de uma operação torne-se igual ao tempo de processamento desta operação e, além de gerar uma restrição de infactibilidade temporal no processador em que ocorreu a alocação, provoca igualmente restrições de factibilidade nas janelas de demanda das operações ainda não alocadas. A propagação destas restrições se dá através de dois mecanismos: Propagação intra-ordem e Propagação inter-ordem, que serão discutidos a seguir.

B.3.1) Propagação intra-ordem

Após a alocação de uma operação, as janelas de demanda das operações antecessoras e sucessoras desta mesma tarefa ainda não alocadas devem ser rearranjadas de forma a compatibilizar as janelas com a restrição tecnológica de precedência inerente ao flow shop.

A propagação intra-ordem no flow shop, pode ser dividida em duas partes:

- Propagação intra-ordem "para trás" e,
- Propagação intra-ordem "para frente".

A propagação intra-ordem "para trás" se ocupa em recortar as janelas das operações predecessoras com base no instante de início da tarefa alocada, fixando, se necessário, novos instantes limites de término das janelas de demanda das operações não alocadas. As janelas de demanda das operações antecessoras ainda não alocadas devem obedecer então à seguinte restrição de factibilidade:

$$DF_{ij}^{\alpha} \leq F_{ik} - \sum_{l=j+1}^{k} t_{ik} \qquad \text{(B.1.5)}$$

onde:
DF_{ij}^{α} = instante mais tarde que a tarefa i deve estar terminada no processador j em qualquer janela de demanda α
F_{ik} = instante de término da tarefa i no processador k resultante da alocação

As janelas devem ser modificadas de acordo com a restrição (B.1.5).

A propagação intra-ordem "para frente" se ocupa em recortar as janelas das operações sucessoras à operação alocada com base no instante de término da tarefa alocada fixando, se necessário, novos instantes factíveis de início das janelas de demanda das operações sucessoras ainda não alocadas. Os instantes de abertura das janelas sucessoras devem obedecer à seguinte restrição de factibilidade:

$$DI_{ij}^{\alpha} \leq I_{ik} - \sum_{l=k}^{j-1} t_{ik} \qquad \text{(B.1.6)}$$

onde:
DI_{ij}^{α} = instante mais cedo que a tarefa i pode ser iniciada no processador j em qualquer janela de demanda α
I_{ik} = instante de início da tarefa i no processador k resultante da alocação

As janelas devem ser modificadas de acordo com a restrição (B.1.6).

Quando estas janelas de demanda são reduzidas pode ocorrer que uma ou várias janelas sejam diminuidas a ponto de terem sua dimensão idêntica ao tempo de processamento. Neste caso propõe-se que, se o número de janelas de demanda da operação for igual a um e a janela tiver sua dimensão idêntica ao tempo de processamento, a alocação deve ser feita independentemente de qualquer outra avaliação. Na Figura (B.1.7) é mostrado um exemplo de alocação decorrente da redução da janela de demanda.

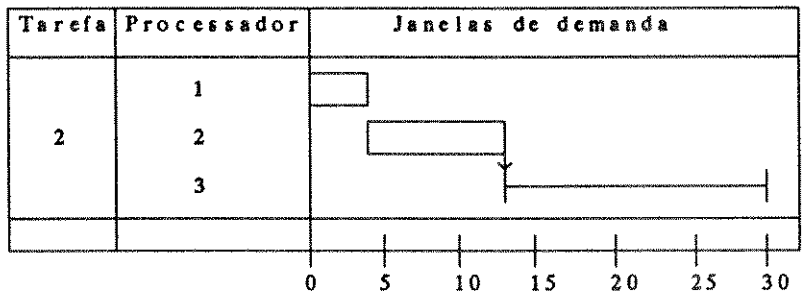


Figura B.1.7 - Alocação resultante da propagação intra-ordem

B.3.2) Propagação Inter-Ordem

A propagação de restrições inter-ordens se ocupa em atualizar as janelas de tempo das demais tarefas como conseqüência da criação de períodos proibidos à alocação do processador para tarefas ainda não alocadas. A propagação inter-ordens se dá em duas etapas:

- Propagação inter-ordem de intervalos proibidos e,
- Propagação nas demais operações de uma mesma ordem dos intervalos proibidos, chamada aqui de propagação intra-ordem de intervalos proibidos.

As janelas de demanda das operações ainda não alocadas não podem conter nem estar contidas total ou parcialmente nos intervalos alocados, quando estas operações demandam os mesmos processadores.

No exemplo em questão foram realizadas duas alocações:
a) Em t=4 até t=13, a operação 2 da tarefa 2 foi programada no processador 2.
b) Em t=0 até t=4, a operação 1 da tarefa 2 foi programada no processador 1.

Estes dois intervalos serão então intervalos proibidos a qualquer alocação futura para as operações não alocadas. Na Figura A.1.8 são apresentadas as propagações das restrições de alocação nas demais operações que requisitam o mesmo processador.

A propagação de intervalos proibidos pode fragmentar as janelas de demanda em duas partes. Cada uma delas deve ser factível, isto é, devem ter uma duração igual ou superior ao tempo de processamento, caso contrário a janela deve ser rejeitada.

A redução das janelas das operações não alocadas deve ser propagada para as demais operações de uma mesma tarefa. Esta propagação é idêntica à propagação intra-ordem já apresentada, devendo apenas ser substituídos os instantes de início e término das alocações que figuram nas restrições (B.1.5) e (B.1.6) pelos instantes de início e término das janelas reduzidas. Na Figura (B.1.9) é mostrada a propagação intra-ordem dos períodos proibidos na operações não alocadas do exemplo em questão.

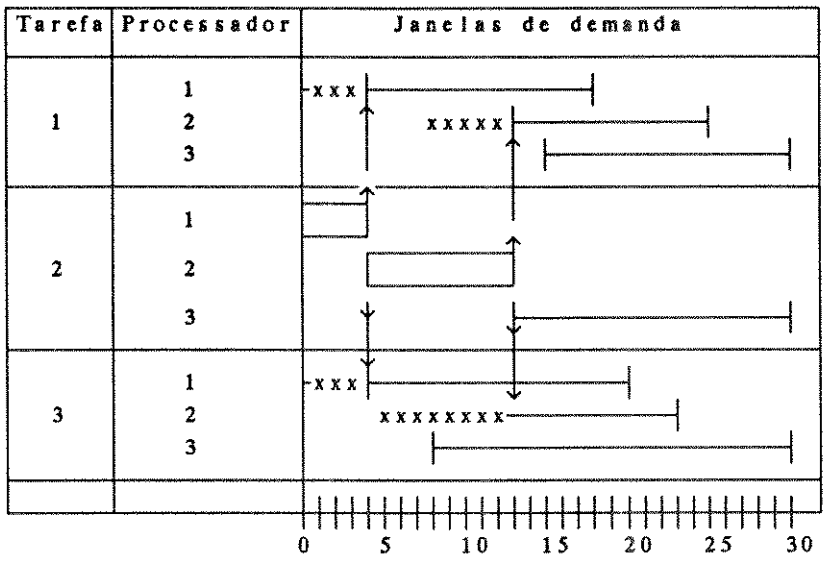


Figura B.1.8 - Propagação Inter-ordens de períodos proibidos

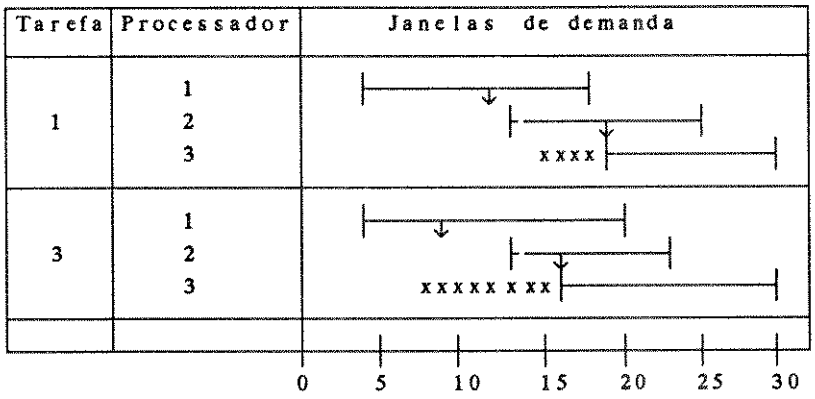


Figura B.1.9 - Propagação intra-ordem de períodos proibidos

Uma vez feitas as alocações e verificada a factibilidade das janelas, o ciclo de ordenamento das operações e alocações deve ser repetido de tal forma que uma nova operação seja escolhida, programada e propagadas as restrições.

A repetição deste ciclo para o exemplo em questão conduz a:
- Maior demanda acumulada: Processador 3
- Intervalo do pico de contenção: [16 a 19]
- Operação mais crítica neste intervalo: Operação 3 da tarefa 3
- Instante de início da operação pela heurística LCV: t=16.

Na Figura (B.1.10) é mostrada a alocação desta operação e a propagação intra-ordem. Como se vê, após a propagação intra-ordem, a operação 2 desta tarefa tem apenas uma janela de demanda reduzida a um intervalo igual ao tempo de processamento. Pela proposta feita anteriormente esta operação deve também ser alocada.

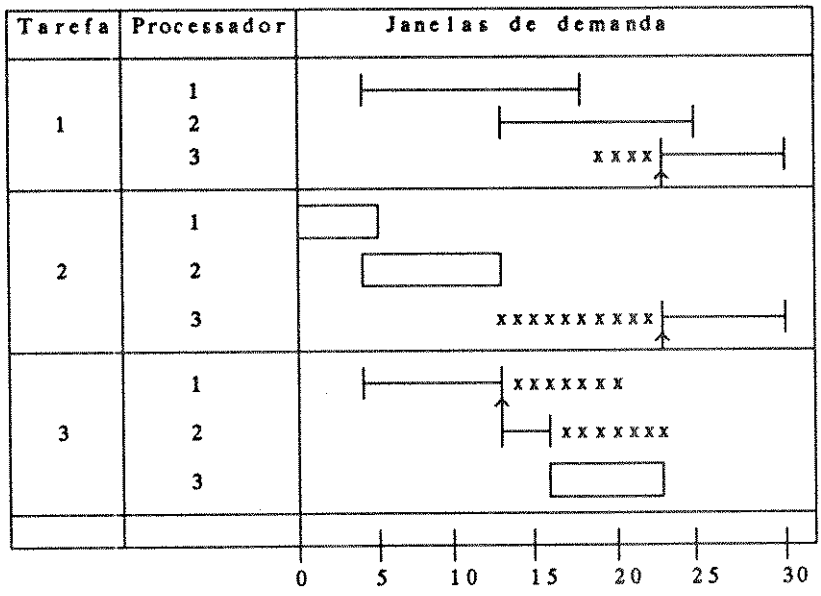


Figura B.1.10 - Segundo ciclo de alocação do exemplo

Na Figura B.1.11 é apresentado o diagrama resultante das alocações feitas até o presente bem como as janelas das operações não alocadas. Uma observação atenta do diagrama apresentado permite verificar que a situação apresentada é infactível, pois impede a construção de seqüências de permutação. De fato, a tarefa 2 é a primeira tarefa da seqüência nos processadores 1 e 2 e, para preservar a seqüência de permutação deveria manter o status de primeira tarefa da seqüência também no processador 3. No entanto a alocação da tarefa 3 no processador 3 decidida através da análise das curvas de criticalidade, impede a contrução da seqüência de permutação.
Este problema na utilização da metodologia de Sycara em flow shops com seqüéncias de permutação tem sua origem no fato que a proposta foi desenvolvida para job shops.

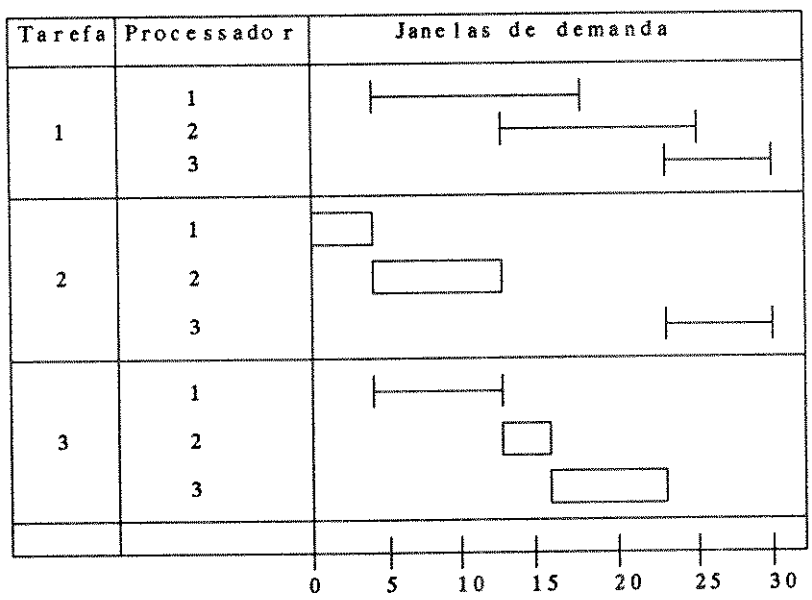


Figura B.1.11 - Diagrama atualizado das alocações e janelas

B.4) Análise de possíveis modificações para aplicação a flow shops

Uma alternativa de preservação das seqüências de permutação seria a definição, nas vizinhanças de uma operação alocada, de uma região proibida para alocações futuras de operações de outras tarefas, tal como mostrado na Figura B.1.12. Note-se que este intervalo de tempo proibido a alocações futuras se propaga "em cascata" a todos os processadores, constituindo de fato uma restrição para a manutenção da factibilidade do fluxo de tarefas no shop. A análise desta figura põe imediatamente a seguinte questão de qual atitude tomar: proibir, através da redução de janelas de demanda das operações não alocadas, os intervalos de tempo adjacentes à operação mais crítica de forma a garantir a manutenção das seqüências de permutação, ou, uma vez escolhida a operação mais crítica, alocar todas as demais operações de uma mesma tarefa.

Na Figura B.1.12 são apresentadas as janelas de demanda que seriam obtidas caso fosse feita a propagação de intervalos proibidos tal como sugerido. Neste caso, no intervalo 13 a 19 o processador 3 seria requisitado exclusivamente pela operação 3 da tarefa 2. Esta operação somente não poderia ser alocada neste intervalo,caso houvessem outros recursos que pudessem, por algum motivo, impedir sua alocação neste intervalo. Em se tratando de um problema de scheduling sem qualquer limitação em outros recursos, a decisão de não alocar a operação 2 da tarefa 3 neste intervalo somente conduziria à inserção de períodos ociosos.Como se deseja como solução uma seqüência de permutação, a inserção de atrasos na execução das tarefas aumenta o makespan. Diante deste quadro a proposta é, uma vez escolhida e alocada a operação mais crítica, todas as operações desta tarefa devem ser alocadas simultaneamente.

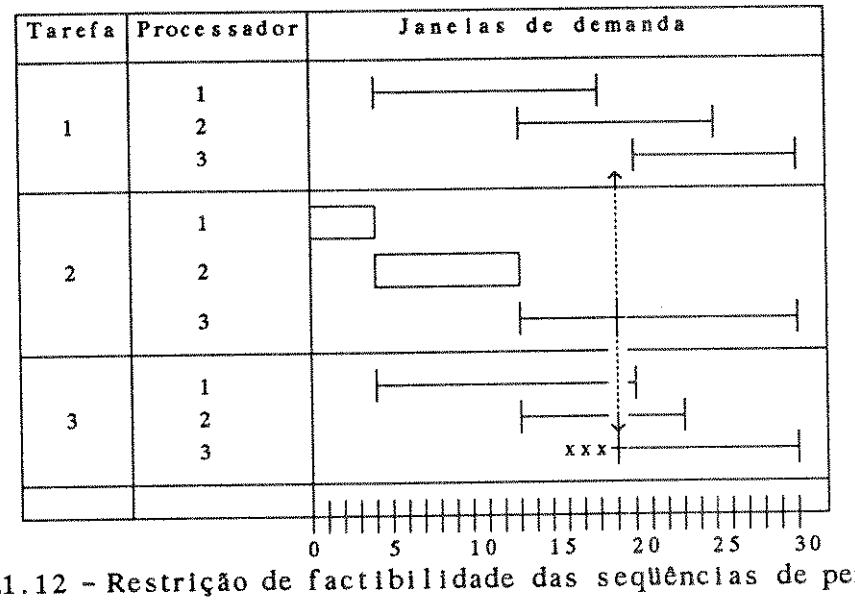


Figura B.1.12 - Restrição de factibilidade das seqüências de permutação

Na Figura B.1.13 é apresentado o diagrama de alocação e as janelas de demanda aplicando a proposta de alocação simultânea de todas as operações de uma mesma tarefa.

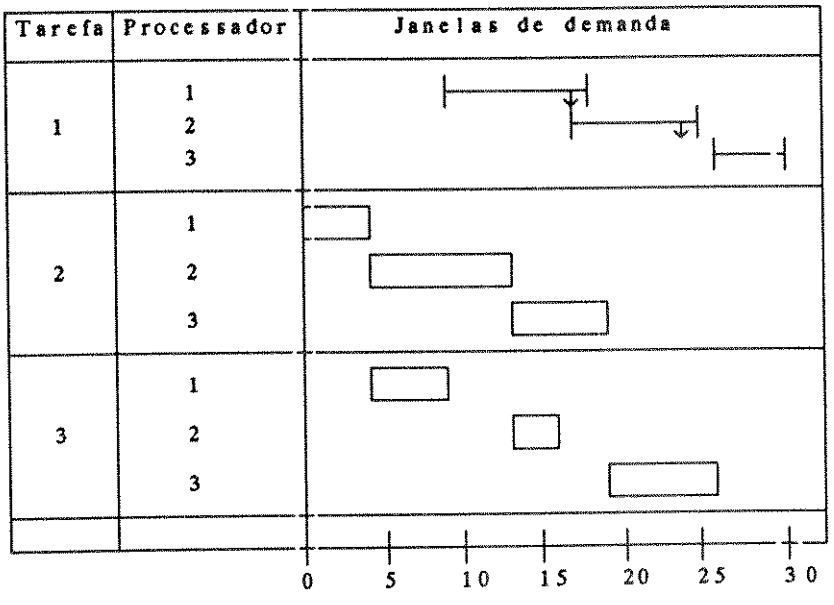


Figura B.1.13 - Diagrama de alocação e janelas de demanda

O resultado prático destas observações é que o conjunto de restrições que devem ser explicitadas para a manutenção da factibilidade tecnológica do flow shop, pode tornar mais vantajosa a alocação simultânea de todas as operações de uma mesma tarefa.

A adoção desta estratégia pode ser utilizada para o seqüenciamento de tarefas, mas seu desempenho comparado ao BAB dependerá da qualidade da heurística empregada na decisão do instante de início da alocação.

Usando a heurística LCV as operações podem ser alocadas no início ou no final da janela demanda, de modo que a dinâmica de funcionamento da metodologia proposta será:

1 - No primeiro ciclo é escolhida a operação mais crítica a ser alocada. Como conseqüência deverão ser simultaneamente alocadas todas as operações desta tarefa.

2 - Se a tarefa mais crítica for alocada no início de sua janela de demanda, automaticamente todas as operações antecedentes à tarefa mais crítica serão alocadas no início de suas respectivas janelas de demanda. As operações posteriores à operação mais crítica deverão ser alocadas o mais cedo possível de forma a reduzir o makespan. Isto equivale a dizer que neste ciclo é escolhida a primeira tarefa da seqüência.

3 - Se a tarefa mais crítica for alocada no final de sua janela de demanda, todas as operações posteriores a ela deverão igualmente serem alocadas no final de suas respectivas janelas de demanda. Com respeito às operações antecedentes podem ser levantadas algumas dúvidas com respeito ao instante em que elas devem ser iniciadas. De fato, as expressões para o cálculo do "completion time" definem que o instante de início de uma operação é definido pelo máximo entre o instante de término da operação antecedente de uma mesma tarefa ou pelo instante de término da tarefa anterior da seqüência no mesmo processador. Neste caso não sendo conhecida ainda a tarefa anterior na seqüência, a política de alocação de menor impacto é equivalente a alocar estas operações o mais tarde possível. Neste ciclo é então escolhida a última tarefa da seqüência.

4 - Nos ciclos subsequentes serão seqüenciadas uma a uma as demais

tarefas não alocadas, não havendo, em princípio, qualquer impedimento em que as tarefas sejam seqüencidas alternadamente no sentido dos tempos crescentes ou decrescentes. Deve-se ter em conta que, usando a heurística proposta, as alocações não serão feitas fora destes dois instantes de modo que as propostas de alocação sugeridas acima são suficientes para guiar as alocações das demais operações em ambos os casos.

A utilização de outras heurísticas de alocação exigirão procedimentos específicos a fim de manter a factibilidade tecnológica da direção de fluxo no flow shop.

No trabalho original os autores propuseram esta metodologia de alocação como uma ferramenta de decisão para ser inserida em um sistema de busca, de forma que, ao ser detectada uma infactibilidade, houvesse o "backtrack". No entanto a concretização desta proposta depende ainda de refinamentos capazes de guiar com maior segurança as decisões de alocação. Os principais pontos a serem ressaltados para investigações posteriores são:

- Quais tipos de infactibilidade podem ser detectadas antecipadamente. No exemplo em questão, após o primeiro ciclo de alocação pode-se verificar imediatamente que, dentro do horizonte de tempo proposto seria impossível alocar todas as tarefas. No terceiro processador verifica-se imediatamente que a janela de tempo máxima disponível é igual a 11 unidades de tempo, enquanto que o processamento das operações não alocadas exige no mínimo 12 unidades de tempo. Esta dificuldade deriva do fato que as curvas de criticalidade não são construídas de forma a revelar infactibilidades "cruzadas".Esta deficiência pode ser exemplificada através da análise da soma das áreas sob as curvas de criticalidade das operações que demandam o mesmo recurso. A área sob a curva de criticalidade reflete a demanda total de uma operação pelo recurso processador, de forma que a soma das áreas das diferentes curvas reflete a demanda total deste recurso. Se esta demanda for maior do que a oferta certamente haverá algum tipo de conflito, no entanto as curvas de demanda por si só são incapazes de fornecer esta informação. No caso em questão, A_{13}= 5 e A_{33}= 7 e a oferta total do processador 3 é A_{P3}=11, sendo A_{ij} a área abaixo da curva de criticalidade da operação i no processador j, e A_{Pj} a área abaixo da curva de demanda acumulada do processador P_j.

- No caso de ser detectada uma infactibilidade são possíveis duas alternativas: factibilizar as janelas de demanda através do relaxamento do horizonte do schedule ou buscar outro caminho de construção da solução. A segunda alternativa exige que a metodologia de alocação proposta esteja inserida em algum tipo de método de busca para que seja definido o nó da árvore para o qual deverá ser feito o retorno. Após o retorno é necessário repetir o ciclo de controle de forma a auxiliar na escolha do novo caminho de construção da seqüência. Ao se fazer o backtracking, a repetição da metodologia de alocação acima conduzirá à escolha do mesmo caminho de alocação, assim a inserção desta metodologia em um método de busca exigirá que seja definida uma política de descarte de caminhos já sondados.

Referencias Bibliográficas

Ashour, S., "An experimental Investigation and Comparative Evaluation of Flow shop Scheduling Techniques",AIIE Trans,2,pp 172,1970

Baker, K.R., Introduction to Sequencing and Scheduling, John Wiley, 1973

Baker,K. R., A Comparative Study of Flow Shop Algorithms, Operations Research, 23(1), pp 62,1975

Bazaraa,M.S., Shetty,C.M. ,"Non Linear Programming Theory and Applications" John Wiley,1979

Bensana, E. et al. ,"An Expert System Approach to Industrial Job Shop Scheduling", IEEE,pp 1645, 1986

Berliner, H., "The B^* Tree Search Algorithm: A Best First Proof Procedure", Artificial Intelligence, 12, pp 23, 1979

Birewar,D.B., Grossmann,I.E., Simultaneous Planning and Scheduling in Multiproduct Batch Plants

Bitran,G.R., Haas,E.A., Hax,A.C., "Hierarchical Production Planning: A Single Stage System, Operations Research",29(4),pp 717,1981

Birewar, B.B., Grosmann, I.E., "Efficient Optimization Algorithms for Zero Wait Schedulling of Multi Product Batch Plants", Ind. Eng. Chem. Res., 28(9),1333, 1989

Booth,F.D., Turner, J.S., "Comparing Flow Shop Scheduling Heuristics", ORSA/TIMS Joint National Meeting, St. Louis , Missouri, October 25-28,1987

Campbell,H., Dudek,R., Smith, R.,"A Heuristic Algorithm for the N Job, M Machine Sequencing Problem",Mgmt. Science,16, pp B630, 1970

Conway,R.W.,et al.,"Theory of Scheduling", Addison Weslwy,1967

Conway,R. et al., "The Role of Work -in-Process Inventory in Serial Production Lines", Oper. Res.,36(2),pp 229, 1988

Coulman, G. A., Algorithm for Optimal Scheduling and a Revised Formulation of Batch Plant Design, Ind. Eng. Chem. Res.,28, 553-556,1989

Dannenbring,D.G.,"An evaluation of Flow Shop Sequencing Heuristics", Magmt. Science, 23(11), pp 11174,1977

Dempster,M.A.H., Fisher,M.L., Analytical Evaluation of Hierarchical Planning Systems, Operations Research, 29(4),707-715,1981

Egli,U.M.,Rippin,D.W.T.,"Short-term Scheduling for Multiproduct Batch Chemical Plants", Comp. Chem. Engng,10(4),pp 303, 1986

Espuña,A., Lázaro,M., Martinez,J.M., Puijaner,L., An Efficient and Simplified Solution to the Pre-Design Problem of Multiproduct Plants, Computers Chem. Engng.,13(1/2),pp 163,1989

Felder,R.M., Simulation - A Tool for Optimizing Batch Process Production, Chemical Engineering,pp 79,1983

Fox,M.S.,"Constraint-Directed Search: A Case Study of the Job Shop Scheduling", Ph.D. Thesis,1987

Garey et al., "The Complexity of The Flowshop and Jobshop Scheduling", Math. Opns. Res.,1,pp 117, 1976

Geoffrion,A.M., "Generalized Benders Decomposition", JOTA, 10(4),11972

Graves, S.C.,"A Review of Production Scheduling", Oper. Res., 29(4), pp646,1981

Grossmann, I.E., Sargent, R.W.H., Optimum Design of Multi-Purpose Chemical Plants, Ind. Eng. Chem. Process Des. Dev.,18(2),343,1979

Grossmann,I.E., Halemane,K.P. , Swaney, R.E., Optimization Strategies for Flexible Chemical Process, Computers and Chemical Engeering, 7(4), 438-462, 1983

Hoitmot,D.J., et al. , "Scheduling Jobs with Simple Precedence Constraints on Parallel Machines", IEEE Control Systems Magazine, pp 34, 1990

Hughes, D.R., Smith,A.W., "Design of an Intelligent Scheduling System", 2nd. Int. Conference on Machine Intelligence, Ed. Prof. A. Pugh., Nov. 1985, London IFS.

Janicke, W., "On the Solution of Scheduling Problems for Multi Purpose Batch Chemical Plants", Comput. Chem. Engng., 8(6),pp 339, 1984

Joglekar,G.S., Clark,S.M., Carmichael, D.S., Batches and Simulation of Multipurpose Plants, AIChE Meeting, Houston,1987

Joglekar,G.S., Reklaitis,G.V., A Simulator for Batch and Semi-Continuous Process, Computers and Chemical Engineering,8(6), 315-327,1984

Kanet, J., Aldsberger, H., "Expert Systems in Production Scheduling", European Journal of Operational Research,29, pp 51, 1987.

Karimi,I., Ku,H., A Modified Heuristic for An Initial Sequence in Flow Shop Scheduling, Ind. Eng. Chem. Res., 27,1654-1658,1988

Keng,N.P.Yun,D.Y.Y.,Rossi,M.,"Interaction-Sensitive Planning System for Job Shop Scheduling",in Expert Systems and Intelligent Manufacturing,pp 57, 1988

Knopf,F.C., Okos,M.R., Reklaitis,G.V., Optimal Design of Batch / Semi - continuous Processes, Ind. Eng. Process. Des. Dev., 21,79-85,1982

Knopf,F.C., "Sequencing a Generalized Two Stage Flow Shop with Finite Intermediate Storage", Computers and Chem. Engng, 9, pp 207, 1985

Korf,R.E., "Planning as Search: A Quantitative Approach", Artificial Intelligence,33, pp 65, 1987.

Kwa,J.,"BS^*: An Admissible Bidirectional Staged Heuristic Search Algorithm",Artificial Intelligence,38,pp 95,1989

Ku,H,Karimi,I.,"Scheduling in Multistage Serial Batch Process with Finite

Intermediate Storage Part 1",AIChE Meeting, Miami, Paper 72e,Nov 1986

Ku,H.,Rajagopalan,D.,Karimi,I.,"scheduling in Batch Process",Chem Eng. Prog,**83**(8), pp 35,1987

Ku, H., Karimi,I.,"Scheduling Algorithms for Serial Multiproduct Batch Processes with Tardiness Penalties",Comp. Chem. Engng,**15**(5),pp 283,1991

Kuriyan,K., Reklaitis,G.V.,"Scheduling Network Flowshops so as to Minimize Makespan", Computers Chem. Eng, 13(1/2),187-200,1989

Kuriyan,K.,Reklaitis,G.V.,Joglekar,G.,"Multiproduct Plant Scheduling Studies Using Boss",Ind. Eng. Chem. Res.,**26**,pp 1551,1987

Lazaro,M., Espuña,A., Puignajer,L., A Comprehensive Approach to Production Planning in Multipurpose Batch Ploants, Computers Chem. Eng.,13(9),1031-1047,1989

Lenstra,J.K.,Rinnooy Kan,A.H.,"Some Simple Applicatios of the Traveling Salesman Problem", Oper. Res. Q.,**26**,pp 717,1975

Lenstra,J.K.,Rinnooy Kan,A.H.,"Complexity of Scheduling under Precedence Constraints",Oper. Res.,**26**, pp 22, 1978

Leinstein,R., "Flowshop Sequencing Problems with Limited Buffer Storage, Int. J. Prod. Res., **28**(11), pp 2085, 1990

Mauderli, A., Rippin,D. W. T., Scheduling Production in Multi-Purpose Batch Plants: The Batchman Program, Chemical Engineering Progress, 76(4), 37-45,1980

Mauderli, A., Rippin,D.W.T., Production Planning and Scheduling for Multipurpose Batch Chemical Plants, Computers and Chemical Engineering,3,199-206,1979

McCurdy,R.,"Applications of Operations Research to Chemical Technology",Ind. Eng. Chem.,**60**(2),pp 14,1968

McMahon,G,Florian,M.,"On Scheduling with Ready Times and Due Dates to Minimize Minimum Lateness", Oper. Res.,**23**(3),pp 475, 1975

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

Nawaz,M.,Enscore,E.,Ham,I.,"A Heuristic Algorithm for the M-Machine, N job Sequencing Problem", The Intl. Jl. of Mgmt Sci.,**11**(1),pp 93, 1983

Nilsson,N.,"Problem Solving in Artificial Intelligence".Mc Graw Hill,1971

Ow,P.,Smith,S.F.,Howie,R.,"A Cooperative Scheduling System",in Expert Systems and Intelligent Manufacturing, Elsevier Science Publishing, Michael D. Oleff(ed),pp 43,1988

Panwalker,S.S.,Iskaqnder,W.,"A Survey of Scheduling Rules", Oper. Res.,**25**(1),pp 45,1977

Pekny, J.F., Miller,D.L.,"Exact Solution of the No-Wait Flow Shop Scheduling with a Comparison to Heuristic Methods",**15**(11),pp 741,1991

Posner,M.E., "The Deadline Constrained Weighted Completion Time Problem: Analysis of a Heuristic", Operations Research,**36**(5),pp 742 1988

Potts,C.N.,"A Lagrangean Based Branch and Bound Algorithm for Singlr Machine Sequencing with Precedence Constraints to Minimize Total Weighted Completion", Mgmt Sci,1978

Rajagopalan,D.,Karimi,I.,"Scheduling in Serial Mixed Storage Product Process with Transfer and Set-uo Times",AIChE Annual Meeting,1987

Reddi,S.S., Ramamoorthy, On the Flowshop Sequencing Problem with No Wait in Process, Operational Research Quarterly,**23**(3), pp 324, 11973

Reklaitis,G.V.,"Review of Scheduling of Process Operations, AIChe Symposium Series", pp 119,1982

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

Rippin,D.W.T., Simulation of Single and Multiproduct Batch Chemical Plants for Optimal Design and Operation, Computers and Chemical Engineering,7(3),137-156,1983

Sacerdoti,E.D.,"Planning in a Hierachy of Abstration Spaces", Artificial Intelligence,**5**,pp 115,1974

Sadeh,N.,Fox,M.S.,"Focus of Attention in an Activity-Based Scheduler", Proc. Nasa Conf. Space Telerobotics,1989

Sahinidis,N.V.,Grossmann,I.E.,"MINLP Model for Multiproduct Scheduling on Parallel Machines,Paper 24c, AIChE Annual Meeting,San Francisco, Nov. 1989

Smith,R. ,Dudek,R. ,"A General Algorithm for the Solution of the N job, M Machine Sequencing Problem of the Flow Shop", Opns. Res., **15**, pp 71, 1967.

Smith,S.F.,Ow,P.S.,"The use of Multiple Problem Decompositions in Time Constrained Planning Tasks",Proc. IJAI,pp 1013,1985

Smith,F.S. et al.,"An Integrated Framework for Generating and Revising Factory Schedules",J.Opl. Res. Soc.,**41**(6), pp539,1990

Sparrow,R.E., Rippin,D.W.T., Forder,G.F., Multi-Batch: A Computer Package of Multi-Product Batch Plants, The Chemical Engineering (London),289,520-525,1974

Sparrow, R.E., Rippin,D.W.T., The Choice of Equipament Sizes for Multi-Product Batch Plants: Heuristics x Branch and Bound, 14(3), 197-203,1975

Steffen, M.S.,Greene,T.J., "An Application of Hierarchical Planning and Constraint-Directed Search to Scheduling Parallel Processors, IEEE, pp 910, 1986

Suhami,I., Mah,R.,"An Implicit Enumeration Scheme for the Flow Shop Problem with No Intermediate Storage", Comp. Chem. Engng,**5**,pp 83,1981

Suhami,I. , Mah,S.H.," Optimal Design of Multipurpose Batch Plants", Ind. Eng. Chem. Process Des. Dev.,**21**,pp 94,1982

Sycara, K. et al.,"Resource Allocation in Distributed Factory Scheduling",IEEE Expert, pp29, fev. 1991

Uskup,E.,Spencer,B.S.,"A Branch and Bound Algorithm for two Stage Production Sequencing Problem", Oper. Res.,23(1),pp 118,1975

Vaselenak, J. A., Grossmann, I. E., Westerberg, A. W., Optimal Retrofit Design of Multiproduct Plants, Ind. Eng. Chem. Res.,26(4),718-726,1987

Vaselenak, J. A., Grossmann, I. E., Westerberg, A. W., An Enbedding Formulation for the Optimal Scheduling and Design of Multipurpose Batch Plants, I.E.C. Research, 26, 139-148,1987

Wellons,M.C.,Reklaitis,"Optimal Schedule Generation for Single-Product Production Line:I Problem Formulation", Comp. Chem. Engng,13(1/2), pp 201, 1989

Wellons,M.C.,Reklaitis,"Optimal Schedule Generation for Single-Product Production Line:II Identification of Dominant Unique Path SequencesProblem Formulation", Comp. Chem. Engng,13(1/2), pp 213, 1989

Wiede,W.,Kuriyan,K.,Reklaitis,G.V.,"Determination of Completion Times for Serial Multiproduct Process 1. A Two Unit Finite Intermediate Storage System",Comp. Chem Eng.,11(4),pp 337,1987

Wiede,W.,Kuriyan,K.,Reklaitis,G.V.,"Determination of Completion Times for Serial Multiproduct Process 2. A Multiunit Finite Intermediate Storage System",Comp. Chem Eng.,11(4),pp 345,1987

Wiede,W.,Kuriyan,K.,Reklaitis,G.V.,"Determination of Completion Times for Serial Multiproduct Process 3. Mixed Intermediate Storage System",Comp. Chem Eng.,11(4),pp 357,1987

Winston,P.H.,"Artificial Intelligence",Addison Wesley Publishing Company,1979

Woollam,C.R.,Sambandam,N.,"The flow Shop Problem with a Composite Cost Function",Comp. Ind. Engng.,9(1),pp 83, 1985

Woollam,C.R.,"Flowshop with No Idle Machine Time Alloowed", Comput. and Indus. Engng., 110(1),pp 69,1986