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

"Uma Abordagem Multi Nível para o Problema do Seqüenciamento de Flowshops com Oferta Limitada de Recursos em Indústrias de Processos Químicos"

Autor : Carlos Alberto dos Santos Passos

Orientador: Prof. Dr. Luis Gimeno Latre
(DCA-FEE-UNICAMP)

Co-orientador : Profa. Dra. Maria Tereza M. Rodrigues
(DESQ-FEQ-UNICAMP)

Tese apresentada à Faculdade de Engenharia Elétrica da Unicamp (FEE-UNICAMP) como parte dos requisitos exigidos para a obtenção do título de DOUTOR em ENGENHARIA ELÉTRICA

Campinas, novembro de 1993

À Rita, e
Isabela.

Agradecimentos:

Ao Prof. Dr. Luis Gimeno Latre pela orientação e pelo excelente nível de relacionamento e amizade.

À Prof. Dra. Maria Teresa Moreira Rodrigues pela co-orientação, amizade e valiosas discussões.

Ao Márcio Dutra pela amizade, discussões e contribuições pessoais ao trabalho.

Aos demais professores e colegas da Faculdade de Engenharia Elétrica e em especial do DCA pelo tratamento a mim dispensado.

Ao apoio da Fundação Centro Tecnológico para Informática sem a qual este trabalho não teria sido realizado.

Ao CNPq pela bolsa concedida para a realização de um estágio de especialização no LAAS dentro do meu programa de doutoramento.

Resumo

Este trabalho trata do problema de seqüenciamento de tarefas em flowshops na Indústria de Processos Químicos com restrições na oferta de recursos de uso compartilhado. A solução proposta para o problema utiliza uma estrutura multi nível onde o problema é dividido em três níveis : pré-seqüenciamento, seqüenciamento e pós-seqüenciamento.

O problema de seqüenciamento é tratado neste trabalho sob duas perspectivas. A primeira através da utilização de um algoritmo que utiliza uma ferramenta sofisticada de otimização, um algoritmo do tipo "Branch and Bound" - BAB no jargão da Pesquisa Operacional ou A* no jargão da Inteligência Artificial, e a segunda através da utilização de um algoritmo de busca heurística guiada por restrições ("Constraint Heurist Search" - CHS) . O BAB permite a otimização de critérios de desempenho global da planta, mas tem o inconveniente de limitar a dimensão dos problemas a serem tratados. A busca heurística ao contrário não garante a otimização de critérios de desempenho globais, mas pode ser aplicado a problemas de grande dimensão. O interessante portanto é combinar estas duas perspectivas para resolver problemas mais próximos dos problemas reais existentes.

A metodologia proposta é implementada através da utilização da técnica de programação orientada ao objeto, utilizando a linguagem C++. Esta técnica se mostrou bastante interessante em função das características da metodologia e da flexibilidade que apresenta em relação à evolução dos programas.

Abstract

This work deals with the problem of flowshop scheduling in Chemical Process Industries constrained by shared resources. A multilevel approach is proposed to solve the problem and it is divided in three main levels: pre-scheduling, scheduling and post-scheduling.

The scheduling problem is solved under two perspectives. The first one utilizes a sophisticated optimization tool, a Branch and Bound Algorithm - BAB in Operational Research or A* in Artificial Intelligence areas, and the other utilizes a Constraint Heuristic Search algorithm. The BAB permits the optimization of global performance criteria's but has the inconvenient to restrict the problem dimension. The heuristic search is the opposite, i.e., doesn't permit the global optimization, but can be applied to problems of high dimension. A good strategy therefore is combine these two perspectives to solve more realistic problems.

The proposed methodology uses an Object Oriented Programming technique, and the C++ language. This technique is interesting because his intrinsic characteristics and the flexibility to the program evolution.

Índice

Introdução 1

1- Planejamento e Programação da Produção 6
1.1 Introdução 6
1.2 O Planejamento e Programação da Produção 6
1.3 Seqüenciamento da Produção 8
1.3.1 Caracterização de sistemas da produção 8
1.3.2 Conceitos básicos 9
1.3.2.1 Definições e objetivos 9
1.3.2.2 A natureza do problema 9
1.3.2.3 Medidas de desempenho 9
A. Critérios baseados em tempos de conclusão 10
B. Critérios baseados em datas de entrega 11
C. Critérios baseados em custos de inventários 11
D. Medidas regulares de desempenho 11
1.3.2.4 Classificação de problemas de seqüenciamento 11
1.3.3 Métodos de resolução 13
1.3.3.1 Pesquisa Operacional 13
1.3.3.1.1 Métodos de otimização 13
A. Métodos específicos 13
B. Programação Matemática 14
C. Métodos arborescentes 14
1.3.3.1.2 Métodos heurísticos 14
A. Métodos heurísticos derivados dos métodos de otimização 14
B. Métodos de construção progressiva 15
1.3.3.2 Inteligência Artificial 15
A. Técnicas de raciocínio baseadas em restrições 16
B. Busca heurística 17
C. Regras e Programação Lógica 17
1.4 Conclusão 18

2 - Técnicas de busca orientada pelas restrições para o *scheduling* de flowshops 20
2.1 Introdução 20
2.1.1 Contexto do seqüenciamento: Preditivo ou Reativo 21
2.2 Definição do flowshop 21

2.3 Elementos numa técnica de busca orientada pelas restrições 23
2.3.1 Processo de busca e backtracking 23
2.3.2 Representação e propagação das restrições temporais 25
A. Janelas de demanda 25
B. Propagação das restrições temporais nas janelas de demanda como resultado de alocações 27
C. Deteção de infactibilidades das janelas de demanda como resultado de alocações 27
D. Caracterização da criticalidade associada a uma operação a partir da sua janela de demanda 29
2.3.3 Representação das restrições sobre recursos compartilhados 30
2.3.4 Utilização de modelos agregados 31
2.3.5. Tratamento do problema através de estruturas hierárquicas e/ou multi nível 32
A. Agregação dos dados 33
B. Categorias de agregação 34
C. Comentários 38
2.4 Técnicas de seqüenciamento baseadas em busca orientada pelas restrições 39
2.4.1 Revisão da literatura 40
A. SRSBP - "Short Range Scheduling for Batch Plants" 40
B. ISIS - "Intelligent Scheduling and Information System" 41
C. OPIS e CORTES 42
D. ISPS - "Interaction Sensitive Planning System" 43
E. DCHS - "Distributed Constrained Heuristic Search" 43
F. MASCOT - "Module d'Analyse Sous Contraintes de Problèmes d'Ordonnancement" 45
G. OPAL - "Système d'Ordonnancement Prévisionel d'Atelier" 46
H. Comentários gerais e características das novas propostas 48
2.4.2 Branch and Bound com busca orientada pelas restrições sobre recursos compartilhados 50
2.4.3 Branch and Bound com busca orientada pelas restrições sobre prazos e recursos compartilhados 55
2.5 Conclusão 59

3 - Algoritmo de seqüenciamento multi nível 60
3.1 Introdução 60
3.2 Descrição do problema 61
3.3 Algoritmo multi nível: Estrutura e elementos 61

3.3.1 Pré-seqüenciamento: Agregação de dados 63
A. Agregação dos consumos 64
B. Agregação dos tempos 65
3.3.2 Pré-seqüenciamento: Classificação dos Recursos 67
3.3.2.1 Classificação baseada na demanda total de recursos 67
3.3.2.2 Classificação baseada na demanda instantânea originada pela oexistência de operações 70
3.3.2.3 Classificação baseada na criticalidade de uma operação 74
3.3.3 Seqüenciamento: Processo de busca orientado pelas restrições 77
3.3.4 Seqüenciamento: Análise de factibilidade 79
3.3.5 Pós-seqüenciamento 81
3.4 Algoritmo multi nível : Descrição global 82
3.4.1 Nível de pré-seqüenciamento 82
3.4.2 Nível de seqüenciamento 85
3.4.3 Nível de pós-seqüenciamento 86
3.5 Observações finais 86
3.6 Conclusão 87

4 - Exemplo de aplicação e apresentação de resultados 89
4.1 Introdução 89
4.2 Comparação entre as soluções obtidas por uma abordagem global e pela abordagem hierárquica no tratamento de recursos compartilhados 89
4.2.1 Exemplos de simulação 91
A. Caso 1: Dois recursos críticos. Análise da escolha total e da escolha isolada de cada um deles. 92
B. Caso 2 : Três recursos críticos. Análise da escolha total e subconjuntos 97
C. Caso 3 : Dois recursos críticos pelo critério T^r e três recursos críticos pelo critério de coexistências. 100
4.3 Apresentação de resultados utilizando o modelo detalhado 103
4.3.1 Exemplo com consumo de recurso idêntico para as três operações 103
4.3.2 Exemplo onde os consumos das operações de preparação e transferência são menores que os respectivos consumos das operações de processamento 108
4.3.3 Exemplo onde os consumos das operações de preparação e transferência são maiores que os respectivos consumos das operações de processamento 112
4.3.4 Análise dos resultados apresentados nas seções 4.3.1, 4.3.2 e 4.3.3 117

4.3.5 Exemplo completo com a utilização da heurística "Beam-search" 117
4.4 Conclusão 125

5 - Conclusões 127

Referências 130

Anexo A - Aspectos da implementação orientada ao objeto 136

Introdução

Objetivo

O objetivo deste trabalho é estudar o problema de seqüenciamento e alocação temporal de recursos em sistemas de produção do tipo flowshop, com ênfase na Indústria de Processos Químicos - IPQ, sujeitos a restrições na oferta de recursos de uso compartilhados (utilidades) e nos instantes de início/fim das tarefas. É proposta uma abordagem multi nível para a solução do problema com o objetivo de tratar problemas com um maior nível de detalhes, e conseqüentemente mais próximos dos problemas reais existentes.

Motivação e discussão do problema

O problema do seqüenciamento de tarefas nas indústrias, embora venha sendo pesquisado a vários anos e até mesmo existam diversos livros que tratem do assunto, não apresenta até o momento uma solução que possa ser considerada satisfatória para a maioria dos casos reais existentes. Um dos fatores que contribui para isto é que as hipóteses utilizadas na maioria das metodologias existentes são muito restritivas, tendo como conseqüência uma limitação na sua aplicação. Um outro fator importante é a diferenciação existente entre os diversos segmentos industriais, o que inviabiliza a existência de propostas genéricas para resolver este problema de forma satisfatória em todos os segmentos. A conseqüência imediata deste último fato é a existência de diferentes propostas para cada um dos segmentos.

Mesmo em segmentos específicos encontramos particularidades inerentes ao tipo de estrutura de processamento que impedem soluções genéricas. Um exemplo típico, é o encontrado nas Indústrias de Processos Químicos , onde existem plantas que são operadas de forma contínua ou descontínua, ou ainda com estrutura de processamento jobshop ou flowshop por exemplo. Conseqüentemente as metodologias utilizadas para tratar os diferentes tipos de problemas na IPQ são também diferentes, pois devem considerar estas particularidades para uma solução mais eficiente.

O problema a ser considerado neste trabalho é o seqüenciamento de tarefas em ambientes de fabricação do tipo flowshop na IPQ, adotando-se uma política de fabricação do tipo UIS ("Unlimited Intermediate Storage"), com seqüências de permutação, sem preempção de tarefas e com restrições na oferta de recursos e datas de início/fim das tarefas.

Um problema freqüente, senão o principal, que se coloca na IPQ é a intensiva utilização de utilidades tais como o vapor e a energia elétrica, que são considerados recursos de uso compartilhado cuja oferta é limitada. Este tipo de problema só aparece e portanto só pode ser resolvido, quando se faz a programação horária das tarefas. Desta forma, o problema tem que ser tratado como sendo simultaneamente um problema de ordenamento e de programação horária, sendo referidos neste trabalho como um problema de seqüenciamento com o mesmo significado do termo em inglês "scheduling".

Este tipo de indústria apresenta certas particularidades que a tornam diferentes do ponto de vista do seqüenciamento, particularidades estas que fazem com que mesmo em problemas de flowshops em áreas correlatas haja uma diferenciação em relação a abordagem utilizada. A principal delas é a baixa flexibilidade da planta ("shop") devido às limitações físicas dos equipamentos, como no caso dos processadores que são interligados através de tubulações com dispositivos do tipo: válvulas, registros e bombas, que para serem remanejados implicam praticamente no reprojeto das instalações.

Outra característica importante é o fato de que quando há uma transferência de produtos entre processadores, ou entre um processador e a armazenagem intermediária, existe uma ocupação simultânea dos equipamentos enquanto durar a operação. Esta característica é normalmente desprezada nas metodologias apresentadas na literatura, principalmente nas que buscam a solução ótima do problema. Esta característica para ser explorada exige que os tempos de transferência sejam considerados independentemente do tempo global das operações. Isto junto com o tempo de preparação faz com que a complexidade do algoritmo seja aumentada significativamente. Como será discutido a seguir, as operações de preparação, processamento e transferência são consideradas individualmente nesta proposta através da utilização de uma abordagem hierárquica com diferentes níveis de agregação dos dados.

Além dos aspectos considerados acima, faz parte da abordagem proposta a existência de janelas para a alocação de tarefas definidas externamente pelo planejador. Estas janelas podem ser originadas, quer seja por características dos produtos, causadas por exemplo devido a instabilidade dos mesmos, ou por requisitos impostos pelos clientes, quer sejam eles internos, i.e., da própria empresa, ou externos.

O seqüenciamento de tarefas como colocado anteriormente, embora corriqueiro na área industrial, não tem na literatura nenhuma proposta de solução ótima. As técnicas existentes utilizam procedimentos heurísticos, que em geral são do tipo regras de

despacho. Os algoritmos A, B, C1, C2 propostos por Rodrigues (1992) e o algoritmo para alocações externas proposto por Campos (1993), ambos enfocando os aspectos relacionados com a utilização de apenas um recurso são os primeiros passos nesta direção.

Os trabalhos na literatura que tratam do mesmo assunto, apenas resolvem o problema parcialmente, i.e., não consideram conjuntamente todos os aspectos citados. Para o caso do flowshop na IPQ com oferta limitada de recursos, o problema é resolvido normalmente em duas fases. Na primeira fase o problema é resolvido considerando-se a oferta de recursos infinita, em seguida na segunda fase a melhor seqüência é submetida a um processo de factibilização onde as tarefas são deslocadas até que a demanda pelo recurso seja compatível com a oferta. Não existe a preocupação de se contrapor o resultado encontrado após a segunda fase com os demais existentes na primeira. Portanto uma eventual seqüência que poderia apresentar um resultado melhor do ponto de vista dos critérios utilisados no seqüenciamento fica assim descartada.

Existem trabalhos em áreas correlatas, como o jobshop por exemplo, que tratam problemas com características semelhantes, utilizando por exemplo o conceito de criticalidade das operações (Keng (1988) e Sycara (1991)), janelas de tempo (Fox (1983) e Egli e Rippin (1986)) e grafos (Esquirol (1987) e Erschler (1976)), mas que apenas tratam do problema de seqüenciamento do ponto de vista local, sem garantir a otimização de nenhum critério de desempenho global.

A solução proposta para o problema, neste trabalho, utiliza uma estrutura multi nível onde o problema é dividido em três níveis: pré-seqüenciamento, seqüenciamento e pós-seqüenciamento. No nível de pré-seqüenciamento são realizadas a agregação dos dados, a classificação dos recursos e a definição dos instantes de início mais cedo e término mais tarde para tarefas que possuam este tipo de restrição. Na fase de seqüenciamento, que também é subdividida em níveis; é realizada no primeiro subnível a geração de seqüências através de um algoritmo de busca utilizando para tal um modelo simplificado do problema e no segundo subnível a factibilização de seqüências através de uma simulação do processo de fabricação utilizando um modelo detalhado. Após o procedimento de factibilização a solução encontrada é contraposta às demais soluções encontradas no subnível anterior para se verificar se ela é a melhor ou não do ponto de vista dos critérios utilizados no seqüenciamento, caso não seja, há um retrocesso ao procedimento de geração de seqüências para que seja retomado o processo de busca. No nível de pós-seqüenciamento são realizados um detalhamento da Carta de Gantt para a seqüência final, uma análise detalhada da utilização dos recursos e a obtenção de

seqüências alternativas que possam eventualmente ser utilizadas caso a primeira se torne infactível por algum novo fato.

O problema de seqüenciamento é tratado neste trabalho sob duas perspectivas. A primeira através da utilização de um algoritmo que utiliza uma ferramenta sofisticada de otimização, um algoritmo do tipo "Branch and Bound" - BAB (Horowitz (1978)) no jargão da Pesquisa Operacional ou A* no jargão da Inteligência Artificial (Nilsson (1971)) e a segunda através da utilização de um algoritmo de busca heurística guiada por restrições ("Constraint Heurist Search" - CHS) (Fox (1983)). O BAB permite a otimização de critérios de desempenho global da planta, mas tem o inconveniente de limitar a dimensão dos problemas a serem tratados. A busca heurística ao contrário não permite a otimização de critérios de desempenho globais, mas pode ser aplicado a problemas de grande dimensão. O interessante portanto é combinar estas duas perspectivas para resolver problemas mais próximos dos problemas reais existentes.

A metodologia proposta é implementada através da utilização da técnica de programação orientada ao objeto, utilizando a linguagem C++. Esta técnica se mostrou bastante interessante em função das características da metodologia e da flexibilidade que apresenta em relação à evolução dos programas.

As contribuições deste trabalho em relação ao estado da arte do problema do seqüenciamento de tarefas em flowshops na IPQ, são as seguintes:

- proposta de uma abordagem multi nível para o problema;
- algoritmos para a classificação e consideração de recursos em função da sua criticalidade;
- proposta de um novo algoritmo de estimativa de limitantes inferiores ("lower bounds") para "Branch And Bound" aplicados a flowshops.

Organização do trabalho

Esta tese esta dividida em cinco capítulos:

No capítulo 1 é apresentada inicialmente uma discussão sobre o planejamento da produção e seus diferentes níveis; em seguida é introduzido o problema do seqüenciamento a partir de uma visão geral do problema, discutindo a sua caracterização, as medidas de desempenho e os métodos de resolução existentes para um problema geral de seqüenciamento.

No capítulo 2 são apresentadas técnicas de buscas orientadas pelas restrições para o problema do seqüenciamento. O capítulo começa com uma discussão sobre o seqüenciamento preditivo e reativo; à seguir são apresentados uma definição para o problema do flowshop e os elementos existentes nas técnicas de busca orientada pelas restrições visando o entendimento das técnicas que serão apresentadas posteriormente. Neste capítulo são também discutidos a utilização de modelos agregados e de abordagens hierárquicas para o tratamento de problemas de seqüenciamento, bem como os algoritmos desenvolvidos por Rodrigues (1992) e Campos (1993).

No capítulo 3 é apresentada a abordagem proposta, discutindo os diferentes níveis existentes na estrutura multi nível. São também discutidas neste capítulo as propostas para a classificação dos recursos e para a agregação dos dados, que servirão para criação de modelos com diferentes níveis de abstração. Neste capítulo é também apresentado o algoritmo C*, uma extensão dos algoritmos propostos por Rodrigues (1992).

No capítulo 4 é realizada um conjunto de simulações que ilustram o funcionamento do algoritmo como um todo para duas situações distintas: i) avaliação do desempenho do algoritmo em função da classificação dos recursos utilizando exclusivamente o modelo agregado e ii) avaliação do comportamento do algoritmo utilizando os modelos agregado e detalhado.

No capítulo 5 são apresentadas as conclusões do trabalho e propostas para trabalhos futuros.

Em anexo são discutidos alguns aspectos relacionados com a implementação dos algoritmos segundo a técnica de programação orientada ao objeto e da implementação em C++.

Capítulo 1 - Planejamento e Programação da Produção

1.1 Introdução

O objetivo deste capítulo é introduzir alguns conceitos considerados importantes para o entendimento e o posicionamento da proposta apresentada neste trabalho dentro do contexto geral do problema do seqüenciamento de tarefas

O capítulo começa com uma definição do domínio do Planejamento e Controle da Produção, no qual o problema de seqüenciamento se insere, discutindo a hierarquia dos diversos níveis de tomada de decisão. A seguir é introduzido o problema do seqüenciamento, iniciando com a caracterização dos sistemas de produção, discutindo os conceitos básicos e os métodos de resolução.

1.2 O Planejamento e Programação da Produção

O domínio do Planejamento e Programação da Produção é um domínio bastante abrangente que engloba diferentes níveis de tomada de decisão numa empresa. No planejamento de produção, de uma forma geral, são considerados aspectos abrangendo desde o capital a ser investido, passando pelas quantidades de produtos a serem produzidas, até a programação horária dos diversos equipamentos. Classicamente são estabelecidos três níveis distintos de planejamento:

1. Planejamento Estratégico - também chamado de planejamento de longo prazo, que em geral considera um horizonte de planejamento de três a cinco anos com discretização mensal. Nesse tipo de planejamento são enfocados os seguintes aspectos:
   - capital necessário para se atingir as metas;
   - valores dos inventários;
   - estimativas de quantidade a produzir e da capacidade de produção;
   - novos produtos a serem lançados;
   - novas tecnologias a serem utilizadas, etc.

2. Planejamento Tático - também chamado de planejamento de médio prazo. Neste caso, o horizonte de planejamento varia de um a dois anos com discretização variando de semanal a semestral. A variação destes períodos se deve principalmente às características da empresa em relação ao tipo de produto e do seu mercado. Os aspectos enfocados no médio prazo são:

• estabelecimento dos níveis de inventário dos produtos acabados em função dos pedidos em carteira mais as previsões de vendas;
• cálculo das necessidades de materiais, com ênfase aos produtos com longo "lead-time", tanto de fabricação como de fornecimento;
• definição de forma grosseira das necessidades de recursos de produção, etc.

3. Planejamento Operacional - também chamado de planejamento de curto prazo. Neste nível, como no anterior, as escalas de tempo para o planejamento são extremamente dependentes das características das empresas, de uma forma geral pode-se dizer que o horizonte de planejamento é de dias discretizados em horas. Em algumas empresas onde os "lead-times" de fabricação são mais longos, o planejamento pode ser feito para períodos de um mês discretizados em dias. A nível operacional são realizados:
• o seqüenciamento da produção;
• a criação das ordens de produção;
• as movimentações de estoque;
• o planejamento detalhado das necessidades de recursos, etc.

A figura 1.1 mostra a relação entre diferentes níveis de planejamento.

A nível estratégico o planejamento é realizado por famílias de produtos e a capacidade de produção em horas/mês. No nível operacional o planejamento é realizado por produtos individualmente e a capacidade de produção em unidades/dia ou unidades/hora. Já no nível intermediário, o planejamento pode ser realizado por famílias de produtos ou por produtos individualmente com capacidade em horas/mês ou horas/semana.

O nível de detalhe a ser considerado no planejamento tem uma forte ligação com o tamanho do horizonte a ser considerado no planejamento. Ou seja, na medida em que se desce na hierarquia de planejamento, o horizonte de planejamento diminui e o nível de detalhe cresce.

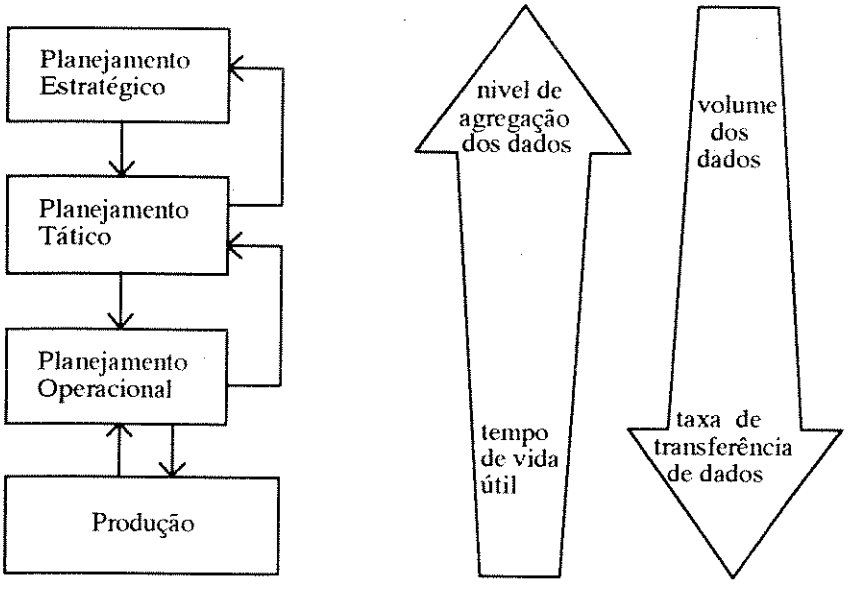


Figura 1.1 : Níveis de planejamento e o sistema de informação

1.3 Seqüenciamento da produção

1.3.1 Caracterização dos sistemas de produção

Os sistemas de produção, do ponto de vista do problema de seqüenciamento, vem sendo caracterizados por diversos autores em diferentes formas (Carlier e Chrétienne (1982), Conway e outros (1967) e French (1982)). A classificação mais comum entre eles é a seguinte:

- flowshop - onde as tarefas possuem uma seqüência de processamento idêntica em todas as máquinas. Em um flowshop genérico máquinas podem ser "saltadas".
- jobshop - onde as tarefas possuem uma seqüência específica, mas podem ser diferentes umas das outras.
- openshop - onde não existem seqüências pré-estabelecidas para as tarefas.

Outros autores propõem classificações mais específicas para os diferentes tipos de sistemas, dentre eles podemos citar Graves (1981) e Baker (1974), que classificam os sistemas a partir de estágios:

- um estágio, um processador
- um estágio, processadores em paralelo
- muti-estágio, flowshop
- muti-estágio, jobshop

1.3.2 Conceitos básicos

1.3.2.1 Definições e objetivos

Graves (1981) define o seqüenciamento da produção como sendo a alocação no tempo dos recursos disponíveis de produção de tal forma a satisfazer da melhor forma possível um conjunto de critérios. O problema do seqüenciamento envolve um conjunto de tarefas a serem realizadas e os critérios podem envolver decisões entre o término mais cedo e/ou mais tarde de uma tarefa.

O objetivo do seqüenciamento é achar uma maneira de atribuir e seqüenciar o uso de recursos compartilhados de tal forma que as restrições de produção sejam satisfeitas e os custos de produção sejam minimizados (Rodammer e White Jr. (1988)).

1.3.2.2 A natureza do problema

Existem diversos fatores que fazem do seqüenciamento um problema difícil. A característica mais proeminente de um problema de seqüenciamento é a sua explosão combinatorial, o que significa, em outros termos, que o número de possíveis seqüências cresce exponencialmente em várias dimensões de acordo com a quantidade de tarefas, operações, máquinas, ferramentas, pessoal, etc. Como exemplo, apresentado por Rickel (1988), o seqüenciamento de 12 ordens com 6 operações tem $(12!)^6$ ou mais de 10^{52} possíveis seqüências em um jobshop sem máquinas alternativas. Esta situação é piorada quando existirem máquinas alternativas que realizem a mesma operação, por máquinas que podem realizar diversas operações e muitos outros fatores. Os problemas de seqüenciamento de flowshops com mais de 3 operações e o de jobshops com mais de duas operações em máquinas distintas, pertencem à classe de problemas classificados como NP-completos (Garey e Johnson (1979) e Rickel (1988)).

1.3.2.3 Medidas de desempenho

Uma medida de desempenho, de maneira geral, pode ser definida como sendo um critério pelo qual podemos julgar o nosso sucesso. Esses critérios, quando aplicados a problemas relacionados com a produção, podem ser definidos matematicamente e de uma forma precisa, a partir de parâmetros relacionados com as tarefas. São eles: r_i - tempo de pronto ("Ready Time"), t_{ij} - tempo de processamento da tarefa i na máquina j, d_i - data de entrega ("Due date") da tarefa i e C_i - tempo de conclusão ("Completion Time") da tarefa i.

A seguir são definidas algumas funções de custo e variáveis utilizadas em medidas de desempenho:

F_i - Tempo de residência ("Flowtime") - Definido como sendo o tempo que a tarefa i gasta no shop, onde : $F_i = C_i - r_i$.

L_i - Defasagem ("Lateness") - Esta função determina a defasagem existente entre o tempo de conclusão e a data de entrega: $L_i = C_i - d_i$. Note que quando uma tarefa estiver adiantada este valor será negativo.

T_i -Atraso ("Tardiness") - Define o atraso de uma tarefa: $T_i = \max\{ L_i, 0 \}$.

E_i - Adiantamento ("Earliness") - Define o adiantamento da tarefa i, onde este valor é calculado por: $E_i = \max\{ -L_i, 0\}$.

I_j - Tempo de ociosidade (" Idle time") - Tempo total que a máquina j ficou ociosa, onde ; $I_j = C_{\max} - \sum_{i=1}^{n} t_{ij}$.

$N_w(t)$ - Número de tarefas esperando entre máquinas no instante t.

$N_p(t)$ - Número de tarefas atualmente sendo processadas no instante t

$N_c(t)$ - Número de tarefas completadas no instante t

$N_u(t)$ - Número de tarefas a serem completadas.

A partir destas funções de custo e demais variáveis é possível definir algumas medidas de desempenho baseadas em critérios do tempo de conclusão, da data de entrega e dos custos de inventário e nível de utilização.

A. Critérios baseados em tempos de conclusão

Os principais critérios baseados em tempo de conclusão são os seguintes:

• F_{max} - Tempo de fluxo ("Flowtime") máximo. Este critério quando minimizado implica que o custo do schedule está associado diretamente com a tarefa mais longa.

• C_{max} - Tempo de conclusão máximo, também chamado de makespan. Minimizando este critério o custo do schedule está associado ao processamento de todas as tarefas.

• $\overline{F}$- "Flowtime" médio. Minimizar $\overline{F}$ implica que o custo médio do schedule está associado ao tempo médio de processamento das tarefas.

• $\overline{C}$- Tempo de conclusão médio. Minimizar $\overline{C}$ é equivalente a minimizar $\overline{F}$. Isto é facilmente verificado, pois como $\overline{C} = \overline{F} + \overline{r}$ e $\overline{r} = \frac{1}{n}\sum_{i=1}^{n} r_i$ é independente da

seqüência e portanto, igual para todas as seqüências. Desta forma, escolhendo um schedule que minimize $\overline{C}$, também estaremos minimizando $\overline{F}$.

B. Critérios baseados em data de entrega

Critérios baseados em datas de entrega, estão sempre relacionados ao cumprimento do prazo de entrega. Este tipo de critério faz sentido somente quando existirem penalidades para o não cumprimento destes prazos. As medidas de desempenho baseadas neste critério mais utilizadas são as seguintes: Atraso médio e máximo, e Defasagem média e máxima. Um outro objetivo relacionado com a data de entrega também existente na literatura é a minimização do número de tarefas atrasadas.

C. Critérios baseados nos custos de inventário e utilização

Os critérios normalmente utilizados neste caso, estão relacionados às tarefas ou às máquinas. Para os critérios relacionados com as tarefas podemos citar os baseados em $\overline{N}_w, \overline{N}_u, \overline{N}_c$ e $\overline{N}_p$, para os critérios relacionados com as máquinas estão o $\overline{I}$ ou I_{max}.

A minimização de $\overline{N}_w$ e $\overline{N}_u$ está relacionada com o custo de inventário em processo, já a minimização de $\overline{N}_c$ com o custo do inventário dos produtos acabados e a maximização de $\overline{N}_p$ está associado ao uso mais eficiente das máquinas. A utilização das funções $\overline{I}$ ou I_{max} também está associada ao uso mais eficiente das máquinas.

D. Medidas regulares de desempenho

Uma medida regular é um valor a ser minimizado que pode ser expresso como função de tempos de conclusão:

$M = f(C_1, C_2, \ldots, C_n)$,

e a qual só aumenta se ao menos um dos tempos de conclusão também aumenta. Desta forma, se $M' = f(C'_1, C'_2, \ldots, C'_n)$, então

$M' > M$ somente se $C'_i > C_i$ para ao menos um i, $1 \leq i \leq n$. Como exemplo de medida regular de desempenho podemos citar $\overline{C}$, C_{max}, F_i, L_i e T_i e como medidas de desempenho que são não regulares podemos citar: $\overline{E}$ e E_{max}.

1.3.2.4 Classificação dos problemas de seqüenciamento

Existe um grande número de diferentes tipos de problemas de seqüenciamento. Esta diferença está relacionada com a quantidade e forma de chegada de tarefas, a

quantidade de processadores, às características da produção, entre outros. Diversos autores na literatura tem utilizado diferentes formas de se classificar estes problemas. Dentre eles podemos citar Conway e outros (1967), Baker (1974), French (1982), Potts (1992), etc. Alguns desse autores apresentam classificação semelhante ou mesmo extensões à propostas anteriores. Uma das classificações que se mostra interessante devido a sua simplicidade e abrangência, é a proposta por Conway (1967), na qual problemas de seqüenciamento são descritos por quatro tipos de informações:

1. As tarefas e operações a serem processadas.
2. A quantidade e o tipo das máquinas que compõem o "shop".
3. Política de fabricação que restringe a forma pela qual as atribuições podem ser feitas.
4. O critério pelo qual o seqüenciamento será avaliado.

Desta forma, a notação utilizada por Conway é representada por quatro parâmetros escritos como A/B/C/D, onde:

- A - descreve o processo de chegada das tarefas. Para problemas dinâmicos, A irá identificar a distribuição de probabilidade dos tempos entre chegadas. Para problemas estáticos, irá especificar a quantidade de tarefas. Se n é utilizado como primeiro termo, ele significará um número arbitrário, mas finito, de tarefas.
- B - descreve a quantidade de máquinas existentes no shop. Um segundo termo m significa um número arbitrário de máquinas.
- C - descreve o modelo de fluxo no shop. Neste caso, F significa um flowshop, R jobshop com fluxo aleatório e G para fluxos arbitrários ou completamente gerais.
- D - descreve o critério pelo qual o seqüenciamento será avaliado.

Exemplos:

1. $n/2/F/F_{max}$ - seqüenciar um número arbitrário de tarefas em um flowshop com duas máquinas minizando o máximo flowtime - problema de Johnson.
2. $n/m/G/F_{max}$ - seqüenciar n tarefas num shop arbitrário de m máquinas na qual a ultima tarefa é terminada o mais cedo possível - problema geral do jobshop.

Seguindo a classificação descrita, o tipo de problema a ser tratado neste trabalho pode ser classificado como $n/m/F/C_{max}$, ou seja, seqüenciar um número arbitrário de tarefas em um flowshop, com um número arbitrário de processadores, no qual a última tarefa deve terminar o mais cedo possível (minimização do "makespan").

O problema objeto deste trabalho envolve também limitações nos recursos de uso compartilhado e restrições de prazo de entrega. Verifica-se portanto a dificuldade de estabelecer uma classificação dos problemas de seqüenciamento abrangente e completa.

1.3.3 Métodos de resolução

Os métodos de resolução para problemas de seqüenciamento podem ser classificados em dois grandes grupos. São eles: os métodos oriundos da Pesquisa Operacional e os métodos oriundos da Inteligência Artificial.

1.3.3.1 Pesquisa Operacional

Os métodos de resolução provenientes da Pesquisa Operacional - PO, são normalmente divididos em dois tipos: os métodos de otimização, também chamados de métodos exatos e os métodos heurísticos. A diferença básica entre eles, é que no primeiro as soluções encontradas são ótimas, enquanto no segundo as soluções não são necessariamente ótimas.

1.3.3.1.1 Métodos de otimização

Este tipo de método procura sempre a obtenção da solução ótima de um problema para um determinado critério. Os métodos de otimização podem ser subdivididos em métodos específicos, métodos oriundos da Programação Matemática e métodos arborescentes.

A. Métodos Específicos

Existem alguns tipos de algoritmos que foram criados para tratar de problemas específicos de seqüenciamento. Estes algoritmos são considerados pertencentes a métodos de otimização quando for possível demonstrar a sua otimalidade. O algoritmo mais conhecido e que se encaixa nesta categoria é o algoritmo de Johnson (Baker (1974) e Dudek(1992)). Johnson propôs um algoritmo para seqüenciar n jobs, todos simultaneamente disponíveis, em um flowshop com duas máquinas o qual minimiza o "flowtime" - $n/2/F/F_{max}$. Um outro algoritmo que pode ser citado é o de Jackson para o caso do seqüenciamento de uma máquina sem restrições sobre o prazo de entrega.

B. Programação Matemática

Através da utilização de um modelo explícito do problema a ser resolvido, é possível utilizar uma ferramenta de uso geral como a Programação Matemática. Os métodos utilizados são do tipo programação linear, não linear e suas variantes associadas, tais como a programação em variáveis inteiras, ou variáveis mistas; mais detalhes podem ser vistos em Papadimitriou e Steiglitz (1982) e Rodrigues (1992).

C. Métodos arborescentes

Estes métodos se encontram dentro do quadro geral da otimização combinatória. A exploração das soluções dos problemas de seqüenciamento se utilizam das técnicas de enumeração implícita, como nos métodos por separação e avaliação progressiva (Ex.: "Branch-and-Bound"). A eficiência destes métodos, depende da função de avaliação e da estratégia utilizada para a exploração da árvore de busca. Uma descrição detalhada destes métodos com aplicação aos problemas de seqüenciamento pode ser obtida em Conway (1967), Baker (1974), Coffman (1976) e French(1982).

1.3.3.1.2 Métodos heurísticos

Nestes métodos, ao contrário do anterior, as soluções encontradas são consideradas "sub-ótimas", ou seja, as soluções encontradas não são necessariamente ótimas. O objetivo é a obtenção de uma solução "razoável" de uma forma mais eficiente do ponto de vista computacional, possibilitando assim tratar problemas de grande dimensão. Estes métodos podem ser classificados em duas categorias; os métodos derivados dos métodos de otimização e os métodos de construção progressiva (French (1982)).

A. Métodos heurísticos derivados dos métodos de otimização

Quando a complexidade do problema (geralmente devido a seu tamanho) inviabiliza a utilização dos métodos de otimização, uma estratégia interessante é a de se "abrir mão" de uma solução ótima em troca de uma "boa solução", onde o método se inspira nos métodos de otimização.

Os métodos de decomposição espacial ou temporal podem entrar nesta categoria. Os subproblemas obtidos após decomposição não são sempre independentes, mas a agregação das soluções locais obtidas podem ser consideradas como uma forma de se aproximar da solução ótima global (Esquirol (1987)).

Podemos citar como exemplo destes métodos heurísticos, aplicados aos problemas de seqüenciamento, as diversas extensões propostas para o algoritmo de Jonhson (Baker (1974)), que é considerado ótimo para o caso de flowshops com duas máquinas, para o caso de flowshops com várias máquinas.

B. Métodos de construção progressiva

Estes métodos tem por princípio a construção progressiva de um seqüenciamento através da simulação do estado de progressão das tarefas no shop. Sempre que houverem conflitos, decisões são tomadas para resolvê-los. Estes conflitos são resolvidos através da utilização de regras de prioridade do tipo: FIFO - "First-In-First-Out", SPT - "Shortest-Processing-Time", LPT - "Least-Processing-Time", MWKR - "Most-Work-Remaining", entre outras (Conway (1967), Baker (1974), French (1982), Zorzo e Cury (1992)).

Os resultados obtidos podem ser o estabelecimento de datas de inicio das tarefas nas máquinas ou o estudo de desempenho das regras de prioridade, do ponto de vista de algum critério, com o objetivo de utilização destas regras como o procedimento de escolha a ser adotado diretamente no "shop", ou seja, podem ser usados num contexto "off-line", no primeiro caso e "on-line" no segundo.

1.3.3.2 Inteligência Artificial

A pesquisa na área de seqüenciamento e de maneira geral na área de Gestão da Produção tem se interessado cada vez mais nos conceitos, metodologias e ferramentas computacionais desenvolvidas na Inteligência Artificial - IA.

A IA exige uma importante reflexão sobre o plano da formalização e da estruturação do conhecimento. Esta reflexão pode levar a uma clarificação e às vezes a um aumento da classe de problemas a serem tratados (Esquirol (1987)).

A maioria da técnicas convencionais de seqüenciamento são baseadas em algoritmos do tipo "Branch-and-Bound" e/ou heurísticas simples. A IA pode ser vista como uma técnica que continua onde a Pesquisa Operacional pára (Rickel (1988)). O fator importante que a IA traz, é a utilização do conhecimento específico do problema, o que sem dúvida permite a obtenção de métodos mais eficientes.

A pesquisa em IA tem sido largamente aplicada na representação e uso do raciocínio heurístico e no desenvolvimento de técnicas de busca mais sofisticadas e eficientes. Apesar das pesquisas em PO e IA estarem centradas nas técnicas de busca, as pesquisas em IA colocam uma maior ênfase no conhecimento que guia as buscas. A seguir serão discutidas algumas das técnicas que são bastante utilizadas.

A. Técnicas de raciocínio baseadas em restrições

Apesar da técnica de raciocínio baseado em restrições ("Constraint Based Reasoning") transcender o escopo da IA, a comunidade de IA tem gasto um esforço significativo de pesquisa nesta área. A contribuição mais importante nesta área, é a utilização de restrições não numéricas. Uma vez representadas, restrições podem ser usadas em várias maneiras. Uma forma de utilizá-las é para determinar a viabilidade de "schedules tentativos" que consiste da geração de sequências que são submetidas a verificação das restrições; se alguma restrição for violada, a seqüência será descartada e uma outra é tentada. Uma forma mais inteligente seria checar as restrições em cada ponto de decisão: um ponto não consistente daria origem a um retrocesso ("backtracking") na busca. Esta ultima abordagem parece ser bem mais interessante que a primeira, mas a combinação das duas não está descartada. A seguir serão apresentados alguns trabalhos da literatura que utilizam ou discutem técnicas do raciocínio baseado em restrições:

Fox (1983) utilizou em seu trabalho "frames" para representar restrições e desenvolveu métodos para "raciocinar" a partir destas restrições. A implementação de "frames" utiliza uma linguagem chamada de SRL ("Schema Representation Language") desenvolvida em Carnegie-Mellon. As restrições utilizadas por Fox, são associadas com os objetos ou ações que elas restringem, facilitando desta forma a determinação das restrições relevantes. Este trabalho será discutido mais detalhadamente no próximo capítulo.

Steffen e Greene (1986) investigaram em seu trabalho a aplicação de métodos de IA para o problema do seqüenciamento de processadores paralelos sujeitos a restrições de preferência, seqüenciamento e inventário em processo. A técnica utilizada consiste da utilização de planejamento hierárquico e busca dirigida por restrições ("Constraint-directed Search"). Um dos motivos que os levaram a utilizar este tipo de abordagem, é que para se gerar seqüências devem ser considerados uma variedade de objetivos e restrições, e desta forma é necessário estabelecer uma relação de compromisso quando as restrições são conflitantes. Um segundo motivo é que a combinação do planejamento hierárquico e busca dirigida por restrições possibilita que problemas de grande dimensão sejam subdivididos

em subproblemas de menor tamanho, facilitando-se a sua resolução. Esta combinação torna, sem dúvidas, este tipo de abordagem atraente.

Na literatura pode-se encontrar revisões recentes das propostas existentes, dentre elas estão as revisões feitas por Atabakhsh (1991) e Le Pape (1991). Atabakhsh (1991) faz um levantamento de sistemas baseados em restrições aplicados aos problemas de seqüenciamento. Neste trabalho é feita uma análise comparativa de diversos sistemas de seqüenciamento bem conhecidos, tais como: ISIS (Fox (1993)), OPIS (Smith (1985)) e SONIA (Le Pape (1991)). Estes sistemas serão apresentados na seção 2.4.1. Le Pape (1991) além da revisão das diferentes técnicas de propagação de restrições faz também uma comparação de resultados experimentais destas técnicas em uma variedade de problemas.

B. Busca heurística

Busca heurística é um termo abrangente que se refere aos procedimentos de busca no qual heurísticas são usadas para guiar as buscas através dos caminhos mais promissores que contenham a solução desejada. Este tipo de abordagem contempla técnicas comuns tais como: "Beam Search" (Fox (1983)), na qual somente os melhores filhos de cada nó são mantidos na busca, algoritmos do tipo A^* (Rich (1983)) que possibilitam a utilização de heurística que conduzem a obtenção da solução ótima e uma infinita variedade de buscas do tipo "best-first", na qual o caminho mais promissor é o próximo a ser explorado.

A pesquisa em IA tem contribuído para a teoria de busca não somente na interminável variedade de buscas heurísticas, mas também na investigação do relacionamento entre heurísticas apuradas e a complexidade correspondente da busca.

Se o espaço de busca é muito grande, mesmo boas heurísticas podem não ser capazes de controlar a complexidade da solução do problema. Um método para controlar isto é através da utilização de uma busca hierárquica no espaço de solução do problema. Uma estratégia a ser utilizada é a estratificação deste espaço, em sucessivos níveis de abstração. Buscando os níveis mais altos em primeiro lugar, decisões podem ser tomadas que limitarão o espaço de busca dos níveis mais baixos (Rickel (1988)).

C. Regras e Programação Lógica

Regras e programação lógica, as quais tem sido utilizadas mais intensamente em sistemas dedutivos, não tem sido muito utilizadas em aplicações de seqüenciamento. Isto

se deve primeiro ao fato de que o seqüenciamento é um problema de natureza combinatorial e segundo porque a maior parte das linguagens baseadas em lógica tem recursos pobres para controlar a busca. Por exemplo, Prolog, o qual utiliza uma estratégia de backtracking cronológico, é um passo certo na direção de falhas em problemas de grande porte. Desta forma, a vantagem do estilo declarativo da programação baseada em lógica é ofuscada pela ineficiência das implementações correntes (Rickel (1988)).

Esquirol (1987) e Rickel (1988) discutem em seus trabalhos uma aplicação, considerada por eles como a mais interessante, de um problema de seqüenciamento jobshop de carros em uma linha de montagem apresentada por Parello (1986). A abordagem utilizada por Parello, segundo eles, implementa um conjunto de heurísticas baseadas em funções de penalidades associadas à colocação dos carros segundo o tipo de opções que devem ser instaladas nos mesmos e em função das opções a serem instaladas nos carros situados na proximidade. O objetivo do estudo é de construir um sistema capaz de cooperar na elaboração de heurísticas adaptadas, facilmente modificáveis, consultáveis à todo momento e dotadas de capacidade de explicação sobre as escolhas efetuadas. A linguagem escolhida para a implementação é chamada de ITP que é uma linguagem de raciocínio automatizado ("Automated Reasoning Language"). ITP é muito parecido com Prolog, a qual junto com OPS5 foram consideradas para a implementação.

1.4 Conclusão

Neste capítulo forma discutidos os diferentes níveis de planejamento envolvendo os aspectos relacionados com o longo, o médio e o curto prazo. Outro aspecto também discutido foi o problema de seqüenciamento, que é tipicamente um problema de curto prazo, e as diferentes técnicas existentes para tratar este tipo de problema. Em relação às técnicas existentes podemos fazer os seguintes comentários:

Os métodos de programação matemática tem como principal inconveniente o fato de que algoritmos mais eficientes que estes podem ser desenvolvidos através da consideração das características especiais do problema, que normalmente estes métodos não viabilizam. Isto é verdade especialmente para métodos do tipo PL, uma das razões é que existe uma grande distância entre a "linguagem do shop" e equações lineares. Este métodos em geral tem ainda a limitação de serem aplicáveis a problemas de seqüenciamento de pequena dimensão.

Os métodos heurísticos, quer sejam oriundos da IA ou da PO, ao contrário dos métodos de programação matemática, são em geral aplicáveis a problema de qualquer

dimensão. Estes métodos heurísticos são bastante eficientes quando as heurísticas são bem adaptadas ao problema, i.e., levam em consideração as especificidades do mesmo, de outro lado quando estas heurísticas são "genéricas", a solução do problema tende a ser pobre comprometendo os objetivos iniciais. Uma diferença básica entre estes métodos e os métodos de programação matemática é que os últimos fornecem soluções ótimas e os métodos heurísticos fornecem soluções nem sempre ótimas.

Em relação ao uso de regras e programação lógica, a conclusão a que se chega, é que elas podem ser interessantes em muitas subpartes do sistema de seqüenciamento, mas até o momento nenhuma aplicação demonstrou as suas habilidades para resolver problemas globais. A maior efetividade na utilização deste tipo de abordagem é em decisões locais, como no caso das regras de despacho por exemplo.

O método de resolução utilizado neste trabalho, e que será apresentado no capítulo 3, é classificado como um método arborescente oriundo da PO combinado com busca orientada por restrições. O algoritmo básico utilizado é chamado de BAB na PO e de A* na IA. O processo de busca é dirigido por uma função de custo a qual calcula o valor para o limitante inferior a partir da satisfação de restrições temporais associadas às operações nos diferentes processadores e restrições associadas a oferta de recursos compartilhados. O critério utilizado é a mimimização do tempo total de fabricação (minimização do "makespan").

Capítulo 2 - Técnicas de busca orientada pelas restrições para o *scheduling* de flowshops.

2.1 Introdução.

O problema de seqüenciamento ("scheduling") em flowshops reais, com todas as restrições presentes e as especificações sobre o resultado desejado, nem sempre traduzíveis em um critério de custo, é difícil de ser representado através de um modelo matemático sobre o qual possa se utilizar uma técnica de otimização. Normalmente, o número de variáveis envolvidas é muito grande e a eficiência computacional da técnica de otimização fica comprometida.

Uma das abordagens que parecem mais promissoras é a utilização de um *método de busca* para a obtenção da solução ótima, ou de uma solução razoável. Os métodos de busca são também pesados computacionalmente, a menos que se utilizem heurísticas que simplifiquem a busca, o que obviamente pode comprometer a solução obtida. O aspecto importante é que as restrições do problema, restringindo a árvore de busca, diminuem o tamanho do problema de busca (Rickel (1988)). Isto, claro, às custas de um esforço na verificação das restrições, processo este que passa a fazer parte do próprio mecanismo de busca.

Em um processo de busca o problema da representação matemática do problema de seqüenciamento aparece de uma forma muito mais simples do que quando se pretende utilizar uma técnica de programação matemática. De fato só se precisa de uma descrição matemática das diferentes condições que representam o processo sem nenhuma necessidade desta descrição ter uma forma especial/particular imposta pela ferramenta de otimização.

Esta descrição matemática do processo deve contudo, seja qual for a técnica de solução que se utilize, representar os aspectos relevantes do processo. Em qualquer problema de seqüenciamento certamente coexistirão aspectos da planta e das tarefas que tem grande impacto sobre a solução do problema e aspectos que tem pouca influência nele. A situação não é estática: para um certo conjunto de tarefas um processador pode ser o gargalo, enquanto que este pode mudar se mudarem as tarefas. Tudo indica portanto que no problema de seqüenciamento sejam aplicáveis metodologias do tipo de agregação de modelos, tratamento hierarquizado ou multi nível e, no aspecto de implementação, a utilização de linguagens orientadas a objetos.

A descrição das técnicas de busca orientada pelas restrições é feita neste capítulo apresentando os seguintes pontos:

- processo de busca e retrocesso ("backtracking");
- representação das restrições temporais em forma de janelas;
- propagação das restrições temporais;
- representação das restrições sobre recursos compartilhados;
- modelos agregados: agregação de tempos e de recursos;
- estruturas multi níveis e/ou hierárquicas.

2.1.1. Contexto do seqüenciamento: Preditivo ou Reativo.

Neste trabalho o seqüenciamento é estudado num contexto preditivo, ou seja, a solução do problema é explorada num contexto "off-line" onde são conhecidos os dados do problema em termos de tarefas, processadores, restrições e critério de custo. A técnica de solução estará sempre baseada num processo de busca.

O seqüenciamento reativo tem como objetivo determinar a próxima tarefa/operação a ser processada levando em consideração as perturbações que afetam o estado atual da planta. Num contexto reativo utilizam-se em geral regras de decisão heurísticas, denominadas regras de despacho, que levam à definição de uma tarefa/operação como sendo a mais conveniente a ser processada. Dado o tempo curto para tomar esta decisão as regras de despacho não consideram o problema de otimização global.

As técnicas de seqüenciamento reativo tem porém características próximas às metodologias heurísticas que às vezes são utilizadas no processo de busca num contexto preditivo. De fato, quando se quer reduzir a árvore de busca no seqüenciamento preditivo o que se utiliza são regras heurísticas que reduzam o número de nós filhos; o seqüenciamento reativo pode ser visto como o limite deste processo quando se define um único nó filho.

2.2. Definição do flowshop.

O problema de seqüenciamento em flowshops que será tratado é definido como sendo o de determinar os instantes de início de cada uma das operações (Carta de Gantt), e é definido da seguinte maneira:

Sendo dados :

- N produtos e suas receitas(processos de fabricação);
- Os prazos de entrega dos produtos;
- A natureza da armazenagem intermediária e a política de processamento;
- A estrutura de processamento e os perfis temporais dos recursos necessários à execução das tarefas e
- O critério de custo ou desempenho do sistema.

Determinar :

- A seqüência de produção;
- A alocação dos equipamentos aos produtos e
- 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 relações de precedência, que definem um ordenamento parcial na execução de tarefas. As restrições tecnológicas de precedência são bastante comuns na indústria química já que freqüentemente alguns produtos são produtos finais e ao mesmo tempo intermediários para a fabricação de outros produtos. Além da informação acerca da interdependência dos produtos, outras informações importantes devem constar das receitas de cada produto tais como :

- Tempo de processamento das operações de uma tarefa;
- Tempo de transferência de produtos entre processadores e armazenagem intermediária caso exista;
- Tempo de preparação dos equipamentos, que pode ser dependente da seqüência ou não;
- Natureza dos produtos intermediários, especificando se o mesmo é instável ou não;
- Relação de recursos necessários a execução de uma operação e o perfil temporal de consumo.

Para o caso da Indústria Química predominam as estruturas de processamento do tipo flowshop, as quais possuem as seguintes características :

- O processamento das operações é feito sem interrupção (sem preempção);
- A ordem de processamento é a mesma em todos os processadores (seqüências de permutação);
- Uso intensivo de armazenagem intermediária;

• Uso intensivo de recursos compartilhados, especialmente utilidades como vapor, água de refrigeração, energia elétrica;
• As conexões entre os equipamentos são limitadas, diminuindo a flexibilidade.

Nos problemas de seqüenciamento os elementos centrais são tarefas e recursos. No entanto, no caso de flowshops onde são exploradas as seqüências de permutação, a direção de fluxo fica fixada pela própria estrutura de processamento, fazendo com que a seqüência de processamento nos processadores seja sempre a mesma, eliminando uma dimensão do problema que é a escolha do processador. Existe porém a demanda simultânea de recursos e, portanto, a possibilidade de geração de conflitos temporais, pelos demais recursos compartilhados tais como : vapor, energia elétrica, etc. Para a construção de programações factíveis é necessário que sejam conhecidos, além dos perfis de demanda que caracterizam as tarefas, também os perfis de oferta que caracterizam os recursos.

2.3. Elementos numa técnica de busca orientada pelas restrições.

2.3.1. Processo de busca e retrocesso.

O processo de busca aplicado a um problema de otimização com restrições pode ser descrito como segue. Trata-se de definir os valores para um conjunto de variáveis x_1, x_2,..., x_n. Para cada uma delas pode se atribuir um valor v_i , $v_i \in D_i$, i=1..n , onde D_i é um domínio definido previamente, sendo que existem relações entre as variáveis que devem ser satisfeitas. A atribuição de valores às variáveis deve minimizar (maximizar) algum tipo de critério em função das variáveis x_i. A utilização de um método exato de busca com retrocesso ("Backtracking Search"), consiste num método de enumeração controlada das soluções possíveis. Partindo de algum ordenamento inicial o método atribui valores sucessivamente às variáveis, dentro do conjunto de valores factíveis e satisfazendo as restrições entre cada atribuição e os valores já atribuídos a outras variáveis. As técnicas de *busca heurística* tentam resolver o problema de dimensão do espaço de busca com uma enumeração parcial que diminuirá este espaço. O problema que se coloca é que a enumeração parcial pode fazer com que a solução ótima não seja encontrada. Consegue-se esta enumeração parcial através da redução do conjunto de valores atribuíveis para as variáveis através de algum processo heurístico e/ou pela redução do espaço de busca.

Seja através de uma busca heurística ou de uma busca orientada pelo melhor, não limitando o espaço de busca, o processo de busca pode chegar a situações em que a solução construída até o momento se torna não factível, ou seja as restrições não podem

ser satisfeitas. Esta situação conhece-se como "dead-end" e o algoritmo deve retroceder eliminando algumas das atribuições já feitas. Isto é o que se denomina "backtracking".

Neste tipo de algoritmo o tamanho da árvore de busca depende fundamentalmente de dois fatores:

- a representação e tratamento das restrições do problema;

  O algoritmo se beneficiará mais das restrições quanto mais explicitamente elas forem representadas, mesmo quando elas resultem da combinação de outras restrições (Rickel (1988) e Fox (1983)). O objetivo será poder levar em consideração o máximo de restrições no processo de busca de forma a diminuir o tamanho da árvore. Obviamente este objetivo terá que ser balanceado com o esforço computacional necessário para mapear todas as restrições em condições a serem testadas em cada nó da árvore.

- a técnica utilizada para a expansão de cada nó da árvore;

  Num extremo, como já foi indicado anteriormente, pode-se utilizar uma técnica que garanta a obtenção da solução ótima, como p.ex. o "*Branch and Bound" com busca pelo melhor*, o que vai levar normalmente a uma árvore grande. No outro extremo, podem se utilizar heurísticas que levem à determinação de um único nó filho para cada nó pai (equivalente a uma *regra de despacho*), a árvore será muito menor mas provavelmente a solução ótima não será encontrada. Entre estes dois extremos tem-se na literatura diversas heurísticas para a expansão dos nós (ex.: "Beam-search" - Fox (1983)) .

No problema de seqüenciamento as variáveis às quais precisam ser atribuídos valores são os tempos de início de processamento de cada uma das operações das tarefas. As restrições diretas sobre estas variáveis são originadas por tempos de pronto ("ready times"), por datas de entrega ("due dates") e por *relações de precedência*. Tem-se restrições indiretas sobre estas variáveis devido à oferta limitada dos recursos compartilhados, necessários ao processamento das tarefas em cada um dos estágios (processadores), que podem restringir os instantes de início do processamento das operações por problemas de disponibilidade de recursos.

Os dois procedimentos inerentes a uma técnica de busca: a seleção de caminhos a explorar a partir de um nó e o retrocesso ("backtracking") quando o caminho leva a um "dead end" são denominados também na literatura respectivamente de "lookahead" (olhar para frente) e "lookback" (olhar para trás). Esta terminologia é interessante porque enfatiza o aspecto de que, na seleção de caminhos pode ser utilizado qualquer mecanismo

preditivo que possa indicar o melhor caminho e, no "backtracking" é importante detetar em que ponto da árvore começaram os problemas. Nenhuma destas duas idéias é fácil de ser implementada (ou tem custo computacional alto). Isto porque o "lookahead" implica em resolver o problema de otimização num horizonte determinado, cuja definição não é obvia e no caso do "lookback" o problema reside em determinar a informação que precisa ser guardada no processo de busca para, após detetar um "dead end", determinar o nó ao qual deve se retroceder de forma a não repetir o erro nem retroceder demais desnecessariamente. Na seção 2.4.2 apresentam-se contribuições para a definição do horizonte de previsão na situação em que se tem restrições sobre os recursos compartilhados. O "backjumping" (Dechter (1990)) é um procedimento que possibilita um retrocesso ao ponto que originou a infactibilidade, sendo portanto uma alternativa que se coloca para a substituição do "backtracking" cronológico.

2.3.2. Representação e propagação das restrições temporais.

As restrições temporais num problema de seqüenciamento aparecem por múltiplas razões. Dentre elas cabe citar:

- tempos de pronto ("ready times");
- datas de entrega de produtos definidas ("due dates" de tarefas);
- intervalos de tempo em que certas tarefas devem ser realizadas, por exemplo no caso de tarefas de execução periódica por razões tecnológicas (limpeza de um processador), ou por política de manutenção de estoques ou porque resultam em produtos intermediários necessários para outras linhas de produção;
- preferência de execução de determinadas tarefas em certos intervalos de tempo originada por características especiais da tarefa, p. ex. consumo alto de um determinado recurso;
- não disponibilidade de um certo processador num intervalo de tempo (ex.: manutenção);
- etc.

A. Janelas de demanda.

Uma forma de uniformizar todas estas restrições temporais é a utilização do conceito de *janela de demanda* por parte de uma tarefa ou operação, definida como o intervalo de tempo entre o instante mais cedo em que a tarefa/operação pode ser iniciada (EBT - "Earliest Beginning Time") e o instante mais tarde em que ela deve estar finalizada (LFT - "Latest Finish Time"). O conceito de janela de demanda tem sido utilizado por vários autores, uma maior formalização para o problema de tratamento e propagação de

restrições encontra-se em (Keng (1988), Sycara (1991) e (Erschler (1986)). Nestes trabalhos estuda-se o problema do jobshop. Para o caso do flowshop a definição de um LFT para uma tarefa i se traduz facilmente em valores de LFT para cada uma das operações devido às relações de precedência. Tem-se:

$$LFT_{i,j} = LFT_{i,j+1} - TP_{i,j+1} \quad ; \quad j = M-1,1 \tag{2.1}$$

onde $TP_{i,j}$ é o tempo de processamento da tarefa i no processador j (operação j da tarefa i) e M é o número de processadores.

De forma análoga a definição de um EBT para uma tarefa define os EBT de cada uma das suas operações.

$$EBT_{i,j+1} = EBT_{i,j} + TP_{i,j} \quad ; \quad j = 1,M-1 \tag{2.2}$$

Os tempos de processamento devem considerar no caso geral os tempos de preparação, transporte, etc.

Quando não existe uma janela de demanda definida para a tarefa pode-se seguir usando este formalismo considerando como EBT o instante atual e como LFT o horizonte de tempo para o qual se está resolvendo o problema de seqüenciamento. Quando este horizonte de tempo não é dado o LFT pode ser definido através de um "upper bound" obtido por alguma heurística.

A propagação de restrições temporais sobre as janelas de demanda tem como resultado principal a redução do espaço de busca. No caso do jobshop, que é o problema estudado em Sycara (1991), não existe nenhuma estrutura de processamento a ser respeitada pela solução fora as restrições de precedência. Já no caso do flowshop e ainda mais no flowshop com sequências de permutação (onde as tarefas seguem exatamente a mesma seqüência em cada um dos processadores), o ordenamento das tarefas e operações está restrito a ser o mesmo em todos os processadores. Isto leva a que neste caso durante a propagação de restrições sobre as janelas deva-se verificar: i) se a redução de janelas leva à definição deste ordenamento e ii) se a redução de janelas implica em infactibilidades porque não respeitam o ordenamento eventualmente já definido como consequência de alocações anteriores (Rodrigues (1992)).

B. Propagação das restrições temporais nas janelas de demanda como resultado de alocações.

A alocação de uma operação no tempo durante o processo de seqüenciamento leva a uma redefinição das janelas de demanda de algumas das operações ainda não alocadas através da propagação de i) restrições sobre tempos de início e fim, ii) restrições de precedência e iii) ocupação do processador alocado. Em Sycara (1991) esta propagação de restrições é separada em dois tipos: *propagação intra-ordem e propagação inter-ordem*, com o termo ordem sendo sinônimo de tarefa. Nesta referência se analisa a propagação de restrições para jobshops considerando como recursos compartilhados apenas os processadores.

Propagação intra-ordem e inter-ordem

A propagação intra-ordem surge como consequência das relações de precedência entre as operações de uma mesma tarefa. Num flowshop consiste na modificação dos EBT das operações posteriores à operação alocada ("forward propagation") devido à definição do instante de termino desta e da modificação dos LFT das operações anteriores ("backward propagation") devido à definição do instante de início da operação alocada.

Há dois tipos de propagação inter-ordem: a propagação da restrição originada pela alocação do processador, que não estará disponível naquele intervalo de tempo para outras operações (de outras tarefas) e a propagação resultante do consumo de recursos compartilhados originado pela alocação da operação. Este consumo pode tornar impossível a execução de outras operações que precisam do mesmo recurso naquele intervalo de tempo.

Se a propagação inter-ordem modifica a janela de demanda de alguma operação é necessário proceder à propagação intra-ordem da modificação na tarefa à qual pertence esta operação.

C. Deteção de infactibilidades das janelas de demanda como resultado de alocações.

Em Erschler (1986) é proposta uma metodologia para detetar relações de precedência num jobshop impostas pelas alocações. Para tanto utilizam-se as janelas de demanda da forma descrita a seguir:

Sejam duas operações (i,j) e (k,l) que não possam ser executadas simultaneamente porque a oferta de algum recurso comum é insuficiente. Cada operação tem uma janela de demanda associada com EBT e LFT. Tem-se o seguinte resultado:

Se $LFT(i,j) - EBT(k,l) < TP_{kl} + TP_{ij}$ então (i,j) precede (k,l) (a)

Se $LFT(k,l) - EBT(i,j) < TP_{kl} + TP_{ij}$ então (k,l) precede (i,j) (b)

As duas situações estão exemplificadas na figura 2.1.

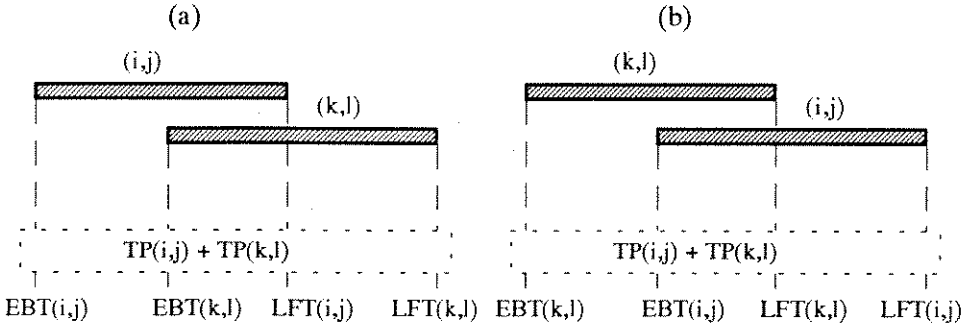


Figura 2.1 : Relações de precedência impostas pelas janelas de demanda.
Caso (a): (i,j) precede (k,l), caso (b): (k,l) precede (i,j).

Considerando uma operação (i,j) e um conjunto de operações {(k1,l1),...(kq,lq)} é possível associar uma operação fictícia (k,l) ao conjunto anterior tal que:

$$TP_{kl} = \sum_{v=1}^{q} TP_{k_v l_v}; \quad (2.3)$$

$$EBT_{kl} = \min(EBT_{k_v l_v}); \text{ e,} \quad (2.4)$$

$$LFT_{kl} = \max(LFT_{k_v l_v}). \quad (2.5)$$

As seguintes relações de precedência podem ser estabelecidas:

Se $LFT(i,j) - EBT(k,l) < TP_{kl} + TP_{ij}$ então (i,j) precede (k1,l1)∪.....∪(kq,lq)

Se $LFT(k,l) - EBT(i,j) < TP_{kl} + TP_{ij}$ então (k1,l1)∪.....∪(kq,lq) precede (i,j)

Estas "relações de precedência" impostas pelas janelas de demanda quando existe conflito na utilização dos recursos comuns podem ser utilizadas no caso do flowshop com

sequências de permutação para detetar infactibilidades, ou seja, "dead-ends" na árvore de busca. Se as janelas de demanda (modificadas dinamicamente pelas alocações sucessivas) impõem uma ordem que não respeita um ordenamento já estabelecido anteriormente o ramo da árvore que se está pesquisando é infactível.

D. Caracterização da criticalidade[1] associada a uma operação a partir da sua janela de demanda.

A janela de demanda de uma operação será maior ou igual ao tempo necessário para o seu processamento. Esta "folga" permite que possam ser atribuídos diferentes tempos de início da operação satisfazendo a sua janela. Em (Keng (1988) e Sycara (1991)) este fato é utilizado para definir uma *criticalidade da operação*, que caracteriza esta "folga". Na primeira referência a criticalidade é definida como o tempo de processamento dividido pelo tamanho da janela, na segunda a criticalidade é função do instante de tempo considerado: dada a unidade de tempo com que se discretiza o problema, calcula-se a probabilidade que cada intervalo unitário de tempo pertencente à janela tem de ser utilizado no processamento da operação (em forma de porcentagem dada pela frequência). Toma-se como criticalidade da operação o valor máximo. As duas definições são semelhantes, em particular a integral da criticalidade sobre a janela é a mesma nos dois casos.

A criticalidade associada a cada intervalo de tempo unitário de cada operação (constante num caso, variável no outro) é utilizada para definir a *crucialidade* de cada intervalo de tempo no que se refere à utilização de cada um dos processadores. Para isto nas duas referências se procede a somar as criticalidades dos instantes de tempo correspondentes a operações que demandam o mesmo processador. Tem-se desta forma uma caracterização dos instantes de tempo do ponto de vista de competição pela utilização dos processadores. Esta informação é utilizada pelos sistemas de seqüenciamento que são propostos nas duas referências.

Nas duas referências anteriormente citadas não se generaliza este conceito de *crucialidade* para a demanda de outros recursos compartilhados além dos processadores.

---

[1] criticalidade, palavra oriunda do inglês "criticality", sinônimo de criticidade, neologismo.

2.3.3 Representação das restrições sobre recursos compartilhados.

Do ponto de vista dos recursos compartilhados o estado do flowshop em qualquer instante de tempo t é caracterizado pelas tarefas que estão sendo processadas, as tarefas ainda não alocadas e o consumo de cada um dos recursos compartilhados. Num processo de busca quando se está analisando um nó da árvore a situação é semelhante: dispõe-se de uma seqüência parcial com as tarefas já alocadas e uma lista de tarefas ainda não alocadas.

A figura 2.2 representa o gráfico que pode ser associado ao instante de tempo t, ou ao nó na árvore de busca. O gráfico mostra a disponibilidade de um recurso compartilhado, suposta constante para simplificar, e o consumo do recurso pelas tarefas já alocadas. Na figura 2.2 para exemplificar se representa a situação com quatro processadores e três tarefas já alocadas (seqüência parcial K = 1-2-3), C(K,1) representa o instante de término de processamento da seqüência parcial K no primeiro processador e C(K,M) o instante de termino de processamento da seqüência no último processador. Estes instantes de tempo permitem definir três regiões: Regiões I, II, e III que tem características diferentes no que se refere à utilização do recurso compartilhado. Na região I, por tratar-se de um flowshop, os recursos não utilizados não poderão ser utilizados por outras tarefas. Já na região II outras tarefas poderão utilizar os recursos ainda disponíveis. Finalmente na região III ainda não existe nenhum comprometimento de recursos.

Em Rodrigues (1992) propõe-se a utilização destas regiões para o cálculo do limitante inferior num algoritmo de busca "Branch and Bound". Os recursos não utilizados na região I implicam num aumento do limitante inferior calculado sob a hipótese de utilização total dos recursos para todas as seqüências parciais sucessoras da seqüência já alocada K. A utilização parcial dos recursos na região II leva a possíveis aumentos do limitante inferior para as mesmas seqüências se a disponibilidade remanescente não permite que a tarefa sucessora inicie o seu processamento em C(K,1).

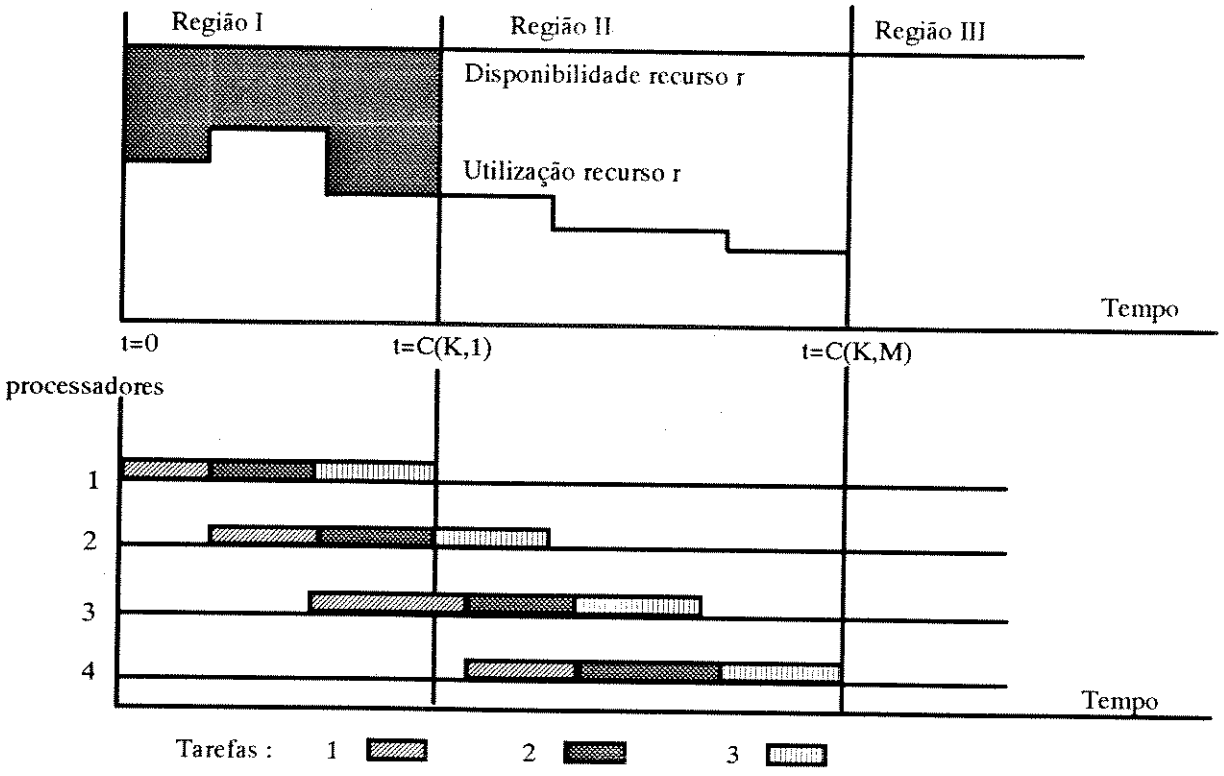


Figura 2.2 : Estado do flowshop quando as tarefas 1, 2 e 3 já foram alocadas

2.3.4 Utilização de modelos agregados.

A utilização de modelos agregados na solução de um problema de seqüenciamento é bastante óbvia. Trata-se de simplificar o modelo da planta e do processamento das tarefas de forma que o algoritmo de seqüenciamento só tenha que trabalhar com as características importantes. De qualquer forma a solução final deverá detalhar todas as etapas do processamento e portanto após a obtenção da solução do problema é necessária uma fase de desagregação e de verificação de que a solução obtida segue sendo factível com o modelo mais detalhado.

Tipicamente são agregados com o tempo de processamento variáveis tais como:

- tempos de preparação ("setup times") quando não são relevantes no problema;
- tempos de transferência entre processadores e entre processadores e a armazenagem intermediária e
- perfis de consumos relativos às operações de preparação, processamento e transferência.

Um outro processo de agregação pode ser feito sobre o perfil de consumo de recursos compartilhados, utilizando-se valores médios que simplificarão o tratamento das

restrições no processo de busca, implicando também numa desagregação a-posteriori para verificar a factibilidade da solução.

Em geral em flowshops complexos algumas tarefas são freqüentemente agregadas na prática porque tem características suficientemente próximas de forma que certamente a solução ótima implica num processamento não interrompido do conjunto.

2.3.5. Tratamento do problema através de estruturas hierárquicas e/ou multi nível.

O seqüenciamento da produção pelo grande número de dados envolvidos e os diversos aspectos do problema que devem ser considerados, faz com que a sua complexidade seja um fator extremamente importante. Neste sentido, duas abordagens diametralmente opostas podem ser utilizadas para se resolver este tipo de problema: a abordagem global e a abordagem hierárquica.

Numa abordagem global o problema é modelado através de um modelo único, englobando todas as variáveis envolvidas no problema. O modelo deve portanto contemplar todos os detalhes que se deseje considerar no problema e tratá-los indistintamente com o nível máximo de detalhe.

Neste caso a quantidade de dados envolvida é bastante grande e levando-se em conta o fato de que na solução do problema deve ser satisfeito um conjunto grande de restrições a tarefa de seqüenciamento fica extremamente "pesada", fazendo com que alguns detalhes tenham que ser desprezados para se garantir que a solução do problema seja obtida em um tempo aceitável.

O princípio da abordagem hierárquica consiste da substituição do problema global por subproblemas que possam ser resolvidos de forma sucessiva e que sejam de dimensão aceitável (Imbert (1986)).

A abordagem hierárquica utiliza os conceitos de agregação e decomposição do problema. A agregação é obtida através do agrupamento de variáveis e a decomposição através da divisão do problema em subproblemas.

As principais vantagens deste tipo de abordagem são as seguintes :

- possibilidade de considerar um universo maior de restrições em um tempo aceitável;

• possibilidade de obter resultados mais detalhados e ,portanto, mais próximos da realidade e

• menor custo de tratamento dos dados.

Este menor custo de tratamento dos dados, se deve principalmente ao fato de que a cada nível é possível ter um tratamento diferenciado dos dados.

Um aspecto importante de abordagens hierárquicas é a coerência das decisões entre os diferentes níveis. A coerência é importante pois as decisões tomadas em um determinado nível fixam os objetivos dos níveis inferiores. Estas decisões normalmente são traduzidas em restrições que são passadas aos níveis inferiores. Desta forma, se uma determinada restrição não puder ser satisfeita em um nível adjacente, é necessário que existam mecanismos que possibilitem o retorno ao nível onde este problema foi gerado e uma nova solução deve ser encontrada. Esta é a forma pela qual a coerência é garantida na abordagem proposta. Maiores detalhes sobre coerência das decisões podem ser obtidos em Imbert (1988) e Dauzere-Peres (1992).

Desta forma, a utilização de abordagens hierárquicas possibilita, em um determinado nível onde os dados encontram-se agregados, resolver um grande número de restrições e o resultado obtido sirva para restringir os demais níveis nos quais os dados estejam mais detalhados. Nestes níveis, é possível então se concentrar na análise de algumas restrições que necessitem de um maior refinamento, ou mesmo, que por algum motivo não tenham sido consideradas nos níveis superiores.

A seguir será apresentada uma breve revisão da literatura sobre a utilização de modelos agregados e uma discussão sobre as diferentes categorias de agregação que podem ser aplicadas a problemas de planejamento, com o objetivo de introduzir os conceitos utilizados na abordagem proposta.

A. Agregação dos dados

A agregação numa abordagem hierárquica pode ser sem sentido caso não se possa desagregar de volta para uma análise mais detalhada dos resultados obtidos com um modelo agregado. Soluções encontradas por este tipo de abordagem são consideradas em geral sub-ótimas, pois o modelo agregado não contempla detalhes normalmente existentes no modelo detalhado. Isto ocorre principalmente pelo fato de que nem todas as possíveis soluções encontradas utilizando-se o modelo agregado serão analisadas detalhadamente e conseqüentemente alguma dessas soluções poderiam ter um resultado melhor, caso fossem

analisadas. Vários autores tem discutido este tipo de abordagem na literatura, entre eles podemos citar : Axsäter (1981), Dempster e outros (1981), Boskma (1982) e Lasserre e outros (1983). A seguir será feito um breve resumo sobre o trabalho destes autores :

Axsäter (1981) estudou o problema da agregação de dados para o planejamento hierárquico de sistemas da produção. A agregação utilizada por ele se concentrou nos dados referentes aos produtos e as diversas máquinas existentes. O objetivo do trabalho era desenvolver diferentes procedimentos para a agregação destes dados referentes aos produtos e máquinas. Axsäter mostra em seu trabalho sob quais condições um planejamento agregado pode ser desagregado de uma maneira consistente.

Boskma (1982) discute em seu trabalho as implicações existentes em relação ao nível de agregação do modelo. Ele discute os modelos que são utilizados no planejamento de médio prazo. Os aspectos considerados por ele são a agregação dos dados relativos aos produtos e aos tempos envolvidos na produção. Uma de suas conclusões é que a agregação de tempos tem que ser aplicada criteriosamente, pois caso contrário, a informação acerca das variáveis endógenas, i.e, as variáveis geradas internamente, poderão ser invalidadas. Na realidade, isto também é valido para as demais variáveis que eventualmente sejam agregadas, quer seja para o tempo ou para qualquer outro elemento.

Lasserre e outros (1983) a exemplo de Boskma também focalizam o seu trabalho no planejamento de médio prazo, tratando da agregação e de métodos de decomposição do problema. A estratégia consiste na utilização de objetivos de manufatura agregados de tal forma a ter o plano baseado em quantidades semanais para todo o horizonte de planejamento de médio prazo. O objetivo desta estratégia é obter um plano de produção baseado num horizonte discreto, utilizando para tal um método de Programação Linear. De acordo com os objetivos semanais definidos no médio prazo, é realizado um seqüenciamento de curto prazo para a primeira semana do horizonte. O artigo apresenta como exemplo de aplicação um caso de fabricação de fotomultiplicadores.

B. Categorias de agregação

Segundo Boskma (1982), a agregação é definida como a forma de abstração pela qual um conjunto de variáveis é substituído por um outro conjunto menor de variáveis de mesma dimensão. Ou ainda, segundo Imbert (1986), a agregação é definida como sendo uma forma de abstração na qual um conjunto de variáveis apresentando certas características comuns, pode ser substituídas por uma variável agregada. A partir destas duas definições é importante ressaltar que a agregação se dá através do agrupamento de

variáveis no sentido de se poder tratar problemas de maior dimensão considerando modelos com maior ou menor nível de abstração.

A agregação de variáveis pode ser classificada em diferentes categorias. Na literatura é possível encontrar diferentes formas de se classificar essas categorias. Dentre elas, a classificação que se mostra mais adequada dentro da ótica do planejamento da produção é a seguinte :

- agregação de características de produtos;
- agregação de recursos, e
- agregação do tempo.

A *agregação de característica de produtos* é obtida através de famílias de produtos que possam ser considerados homogêneos em relação a características relevantes. Essas famílias podem ser criadas a partir da sua composição básica, do processo de fabricação, ou a partir de uma mesmo tipo de demanda, entre outras.

Como exemplo para este tipo de categoria podemos citar uma estrutura com dois níveis diferentes : agregado e detalhado. No nível mais agregado os produtos podem agrupados por tipo, figura 2.3.

Figura 2.3 : Agrupamento de produtos

Neste caso, o planejamento no nível agregado é feito levando-se em conta apenas o tipo de produto (caneta e borracha) e no nível mais detalhado os diferentes artigos (caneta tinteiro, caneta esferográfica, borracha para lápis, etc). Desta forma, como citado anteriormente, os resultados que venham a ser obtidos no nível agregado, devem ser desagregados e submetidos a uma análise detalhada para a elaboração do plano final.

A *agregação de recursos* é obtida a partir de características funcionais dos recursos. Alguns dos tipos de características funcionais que podem ser agregadas são as seguintes : máquinas que realizam operações semelhantes, trabalhadores que realizam as mesmas tarefas, fontes de energia com custos semelhantes, etc.

A forma mais simples de agregação de recursos e mais mencionada na literatura, é a agregação de equipamentos idênticos, como mostrado na figura 2.4.

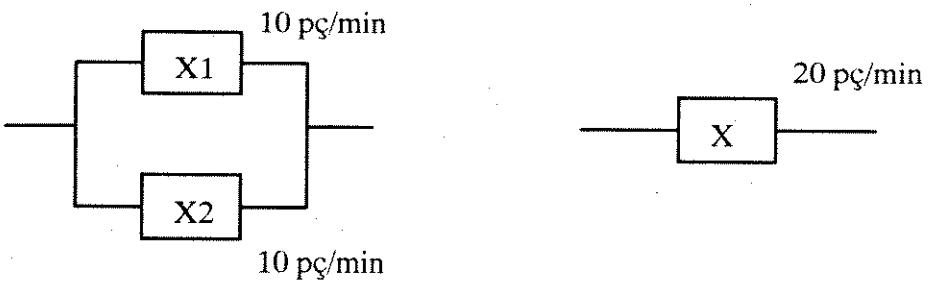


Figura 2.4 : Agregação de equipamentos idênticos

Neste caso, a capacidade nominal dos equipamentos é agrupada e os recursos são considerados de forma agregada. Mesmo neste caso mais simples, a desagregação se faz necessária, principalmente em condições nas quais se deseje uma análise detalhada da utilização dos recursos, como no caso de programação para manutenção preventiva por exemplo.

A *agregação do tempo* é obtida através da utilização de diferentes escalas de tempo nos diferentes níveis da estrutura de planejamento . Um nível mais detalhado trabalha com uma escala de tempo mais refinada, enquanto que um nível mais agregado trabalha com escalas de tempo mais grosseiras. Desta forma períodos em níveis mais agregados correspondem a agrupamento de períodos de níveis detalhados.

Outra característica interessante é a possibilidade de se trabalhar com diferentes horizontes de planejamento a cada nível. A figura 2.5 mostra de forma esquemática esta situação.

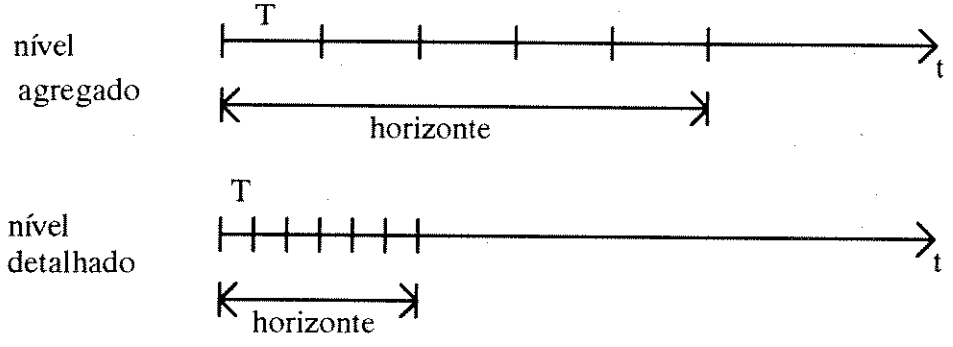


Figura 2.5 : Horizontes de planejamento

Os diferentes níveis de abstração de uma estrutura hierárquica de planejamento são obtidas em função da quantidade de variáveis e equações do modelo. A abordagem hierárquica se aproxima do modelo real de maneira mais fina e detalhada, a medida que se desça na estrutura de planejamento.

Um detalhe importante na agregação dos dados é a consistência entre os diferentes níveis. Caso haja inconsistência entre os níveis, a solução obtida em um determinado nível pode fazer com que as restrições dos demais níveis não possam ser satisfeitas, e portanto seja impossível encontrar uma solução para o problema. Em Imbert (1986) são apresentados alguns tipos de inconsistências decorrentes de agregações baseadas em produtos e no tempo.

A seguir é apresentado um exemplo tirado de Imbert (1986) o qual apresenta um caso de inconsistência entre resultados obtidos em dois níveis de planejamento.

Sejam dois artigos agrupados em um mesmo tipo e para o qual será feito o planejamento das quantidades a serem produzidas considerando um horizonte de dois períodos. E sejam :

d_{kt} : demanda do artigo k no período t

x_{kt} : produção do artigo k no período t

s_{k0} : estoque inicial artigo k

D_t : demanda do tipo para o período t (variável agregada)

X_t : produção do tipo para o período t (variável agregada)

S_0 : estoque inicial do tipo (variável agregada)

As demandas para cada artigo são resumidas na seguinte tabela

d_{kt}	t=1	t=2
k=1	20	10
k=2	10	20

Os estoques iniciais são dados por : S_{10}= 10 e S_{20}=20. Agregando estes dados teremos:

$D_1 = d_{11} + d_{21} = 30$

$D_2 = d_{21} + d_{22} = 30$

$S_0 = s_{10} + s_{20} = 30$

A demanda D_1 pode ser satisfeita pelo estoque existente e uma possível solução agregada será portanto :

$X_1 = 0$; e $X_2 = 30$

Esta solução, embora consistente no nível agregado, não permite uma desagregação factível da demanda de cada artigo. A única desagregação possível para $X_1 = 0$, é :

$x_{11} = 0$; e $x_{12} = 0$.

Conseqüentemente a demanda d_{11} não pode ser satisfeita, enquanto restarão 10 unidades do artigo 2 em estoque após a satisfação da demanda.

C. Comentários

Diversos trabalhos tem sido publicados na literatura que tratam do problema do planejamento hierárquico, discutindo as suas vantagens em relação às abordagens globais. Alguns destes trabalhos já foram aqui discutidos ou mencionados, outros também o serão mais adiante, mas um deles Sacerdoti (1974), que embora trate do planejamento hierárquico de uma forma geral, serviu de inspiração inicial para a utilização de uma estrutura hierárquica neste trabalho.

Sacerdoti (1974) utiliza o conceito da representação de problemas através da utilização de uma hierarquia de espaços abstratos, nos quais são introduzidos sucessivamente um maior nível de detalhe. Ele utiliza o sistema ABSTRIPS que é uma extensão ao sistema STRIPS para resolução de problemas de princípio geral, o qual utiliza regras heurísticas para realizar as buscas.

Neste trabalho é apresentada uma abordagem para incrementar o processo de busca heurística. A essência da abordagem é a utilização de um mecanismo que faz a discriminação entre informações importantes e detalhes no espaço do problema. O planejamento numa hierarquia de espaços abstratos no qual sucessivos níveis de detalhes são introduzidos, possibilitou um aumento significativo da capacidade de resolver problemas.

Para clarificar os conceitos apresentados Sacerdoti utiliza um exemplo de aplicação no qual um robô deve ser conduzido de um estado inicial, localizado numa determinada sala do laboratório, até um estado final, uma outra sala na mesma dependência, alterando

antes o posicionamento de duas caixas que se encontram numa terceira sala, evitando todos os obstáculos do caminho, inclusive portas fechadas.

Para justificar a abordagem, a qual utiliza espaços abstratos, é feita uma comparação entre o espaço de busca gerado quando se utiliza o sistema STRIPS e quando se utiliza o sistema ABSTRIPS. O resultado final apresentado é que a utilização de espaços abstratos melhoram significativamente o desempenho do sistema.

Outro autor que explora a utilização espaços abstratos é Wilkins (1987). Ele explora a utilização de nível de abstração e nível de planejamento, através da mistura destes dois conceitos numa abordagem hierárquica. Segundo ele a essência do planejamento hierárquico é o uso de diferentes níveis de abstração no planejamento do processo e na descrição do domínio. Wilkins apresenta um sistema chamado de SIPE e discute a sua utilização usando um exemplo semelhante ao de Sacerdoti. Um dos aspectos mais interessantes discutido por ele é a confusão existente na literatura que utiliza o termo planejamento hierárquico não somente para descrever níveis de abstração, mas também para descrever sistemas contendo várias estruturas hierárquicas ou espaços de busca, meta níveis, etc. Estes últimos segundo ele deveriam ser chamados de níveis de planejamento.

A proposta deste trabalho contempla segundo a definição de Wilkins uma estrutura composta por níveis de planejamento (pré-seqüenciamento, seqüenciamento e pós-seqüenciamento) e no nível de seqüenciamento uma estrutura hierárquica com dois níveis onde cada um deles possui um diferente nível de abstração.

2.4 Técnicas de seqüenciamento baseadas em busca orientada pelas restrições.

Nesta seção apresentam-se inicialmente algumas das técnicas existentes na literatura baseadas em processos de busca orientada pelas restrições. A revisão da literatura não pretende ser completa, discutem-se as propostas próximas ao problema que esta sendo estudado e as propostas que utilizam metodologias do tipo das que serão propostas no algoritmo apresentado no capítulo 3. Na literatura é possível ainda encontrarmos outros trabalhos que tratam do mesmo tipo de problema tratando-os através de técnicas de Programação Matemática do tipo Programação Linear, Não Linear, Inteira, Inteira Mista, Grafos, etc. Dentre eles podemos citar os artigos de Kondili e outros (1993) e Janiak (1988). Estes últimos não serão aqui discutidos.

2.4.1 Revisão da literatura.

A. SRSBP - *"Short Range Scheduling for Batch Plants"*.

O sistema proposto por Egli e Rippin (1986) realiza o seqüenciamento de curto prazo de uma planta química multiproduto com produção em lotes. Uma característica interessante deste sistema e que vem ao encontro da proposta deste trabalho, é a consideração de recursos de uso compartilhado com limitação na oferta, o que não é comum na literatura.

O algoritmo utilizado na solução do problema consiste de nove estágios. O primeiro deles é responsável pela construção de grupos de produtos independentes, de tal forma que diferentes grupos não tenham produtos intermediários e equipamentos sejam atribuídos somente para produtos pertencentes ao mesmo grupo e, desta forma, o seqüenciamento é realizado para cada grupo independentemente. O segundo estágio consiste na definição de uma lista de prioridades. Esta lista tem a finalidade de definir qual produto deve ser escolhido no procedimento de enumeração no estágio 4. O estágio 3 é responsável pela definição de blocos estáveis, formados pelas operações que podem ser interrompidas em meio a um processo e retomadas posteriormente. Estes serão os blocos que poderão ser deslocados ao longo do calendário para satisfazer as restrições dos estágios 6 e 7. No estágio 4 é realizado o procedimento de enumeração das seqüências factíveis de produção, sem no entanto considerar as restrições sobre os recursos e o tempo, e a política de armazenagem, que serão consideradas nos estágios a seguir. O estágio 5 consiste de um procedimento de reprogramação das tarefas deslocando o seu início para o instante mais tarde possível com o objetivo de diminuir os estoques de produtos acabados. No estágio 6 é realizado um procedimento de factibilização das restrições na oferta de recursos e da jornada de trabalho, deslocando as tarefas em direção do instante de início mais cedo. O estágio 7 é responsável pela verificação da existência de matéria prima, atrasando as tarefas se necessário. No estágio 8 é feito um levantamento do custo da seqüência que está sendo analisada e uma comparação com o menor custo já obtido, caso este custo seja maior a seqüência é descartada. Após este procedimento o programa retoma o procedimento de enumeração. O ultimo estágio, estágio 9, recupera após o término da enumeração a melhor seqüência e o seu custo para uma apresentação dos resultados.

A determinação da melhor seqüência é obtida a partir da enumeração de todas as possíveis seqüências obedecendo a lista de prioridades, o que restringe sobremaneira a

aplicação deste programa a problemas de pequena dimensão. Os autores propõem uma estratégia de agrupamento dos produtos que possam ser tratados independentemente (estágio 1), o que faz segundo eles, que o problema seja decomposto e portanto haja uma redução substancial da magnitude do problema combinatorial. Mas, quando o problema tiver restrição na demanda total por utilidades ou outros recursos devem ser considerados, o que seria o caso geral, o agrupamento de produtos não pode ser considerado independente e, portanto, o agrupamento não pode ser utilizado. O programa está dimensionado para um máximo de 6 produtos com duas variantes no processo, 20 módulos de equipamentos e um período de planejamento máximo de 60 dias.

B. ISIS - *"Inteligent Scheduling and Information System".*

O domínio do problema de seqüenciamento objeto do sistema ISIS (Fox (1983)) é bastante complexo, pois considera uma grande número de fatores, que por vezes, são conflitantes. Estes fatores estão relacionados com o prazo de entrega, restrições de custo, nível de produção, capacidade das máquinas, processos alternativos, características das ordens e dos recursos, e disponibilidade dos recursos. Para tratar desta complexidade, a proposta do ISIS se baseia em duas ideias :

- uma formulação explícita das influências destes aspectos como restrições na base de conhecimento do sistema; e
- a formulação da construção das sequências como uma busca heurística dirigida pelas restrições.

O primeiro ponto acima presume uma visão abrangente das restrições. É importante notar que a representação das restrições no ISIS engloba os objetivos, metas e preferências do seqüenciamento, bem como, uma série de condições necessárias que definem o espaço admissível do "schedule". Restrições provêm a base para a otimização durante a avaliação de soluções alternativas, atribuindo índices que indicam o grau de satisfação das restrições. A representação também captura outros conhecimentos necessários para efetivamente raciocinar com as restrições, incluindo a importância, relevância e interdependência das restrições.

A geração das seqüências na planta é realizada de uma maneira incremental. Para cada ordem a ser seqüenciada, o sistema prossegue através de múltiplos níveis de análise, utilizando uma busca heurística, a qual realiza uma seleção detalhada das operações, dos recursos e dos intervalos de tempo para produzir uma ordem. Trabalhando a partir de um conjunto admissível de roteiros para as ordens, a busca prossegue de traz para a frente a partir da data de entrega ou ao contrário a partir da data de início requerida para a ordem.

O espaço de busca é composto de estados que representam seqüências parciais alternativas, e sobre eles são aplicados operadores de busca que servem para gerar novos estados que incrementarão a descrição de seqüências parciais em desenvolvimento (e.g. acrescentar uma nova operação à seqüência parcial, fixar uma determinada máquina à uma operação, etc.). Usando "Beam-Search", somente os n melhores estados de busca são expandidos em cada iteração. A saída desta busca é um roteiro específico para uma ordem, com a especificação de limites de tempos para os recursos requeridos para o processamento das ordens. Estas definições servem para adicionalmente restringir qualquer seqüenciamento que venha a ser realizado posteriormente.

C. OPIS e CORTES.

O sistema OPIS ("Opportunistic Intelligent Scheduler") (Smith (1985) e (1990)) tem sua origem no ISIS. Este último utiliza uma decomposição baseada em tarefas que agrupa as restrições que concernem uma determinada ordem/tarefa e que permite resolver adequadamente os problemas do seqüenciamento originados por conflitos na tarefa. No sistema OPIS propõe-se uma decomposição também baseada em conflitos por recursos, que implica num agrupamento de restrições que permite resolver problemas do seqüenciamento relativos a várias tarefas. A administração destas duas perspectivas de decomposição do problema é feita através da deteção de gargalos em recursos, a partir da segunda perspectiva, levando a alocação de recursos críticos às tarefas envolvidas. Os recursos não críticos são alocados utilizando a outra decomposição baseada em tarefas. Como colocam os autores, as alocações de recursos críticos a operações criam ilhas de certeza (do ponto de vista que são soluções factíveis) a partir das quais se trabalha uma estratégia baseada nas tarefas.

No sistema CORTES (Sadeh (1989)) desenvolvido pela mesma equipe a análise é feita a nível de operações, eliminando o problema de que, detetada uma operação que constitui um gargalo, toda a a tarefa à qual pertence a operação seja rotulada como crítica. Neste sentido o sistema é caracterizado como sendo uma busca micro-oportunísta.

O processo de busca nos dois sistemas utiliza processos heurísticos para determinar o caminho de busca. Uma versão distribuída (multi agente) do segundo sistema é apresentada em (Sycara (1991)) e vai ser descrita na seção 2.4.1.E.

D. ISPS - *"Interaction Sensitive Planning System".*

Neste sistema proposto em (Keng (1988)), o jobshop original é decomposto em um conjunto de subproblemas com base no horizonte de tempo disponível para processar cada operação, horizonte este dado pela janela de demanda da operação. As janelas de demanda são utilizadas não só para restringir o espaço de busca e manter a consistência das restrições do problema, através da propagação de restrições, mas também para obter a solução ao problema utilizando heurísticas que ordenam as operações a serem alocadas em função da sua criticalidade.

O sistema ISPS está baseado em quatro módulos:

1. *Seleção da operação com maior criticalidade*: utilizando a definição de criticalidade de uma operação, apresentada na seção 2.3.2-D, escolhe-se como a operação a ser alocada aquela que for a mais crítica.
2. *Determinação da crucialidade dos intervalos de tempo*: para cada um dos intervalos unitários de tempo da janela de demanda da operação determina-se a crucialidade (seção 2.3.2-D), levando-se em consideração as operações já alocadas.
3. *Alocação da operação*: utiliza-se uma heurística do tipo menor impacto que aloca a operação naqueles intervalos de tempo onde a crucialidade é menor. Desta forma tenta diminuir a competição pelo processador naqueles intervalos de tempo com maior crucialidade por parte de operações ainda não alocadas.
4. *Propagação das restrições*: a alocação da operação restringe o uso do processador e implica em modificações das janelas de demanda das outras operações ainda não alocadas. Estas restrições são propagadas (seção 2.3.2-B). Se é detetada alguma infactibilidade o sistema ativa uma função denominada "*contradiction handling*" que desfaz alocações já realizadas buscando uma nova solução ("backtracking").

E. DCHS - "Distributed Constrained Heuristic Search"

A exemplo de Keng, embora de uma forma diferente, Sycara (1991) também utiliza no sistema DCHS, o conceito de "criticalidade". Sycara calcula o valor da criticalidade a partir de uma função de verossimilhança associado a ocupação de um processador. 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 de tempo e divide-se pelo número total de alocações possíveis.

A figura 2.6 compara as duas definições de criticalidade para um caso particular onde o DW_j (janela de demanda para a operação j) = 6 u.t. e e_j (tempo de processamento da operação j) = 3.

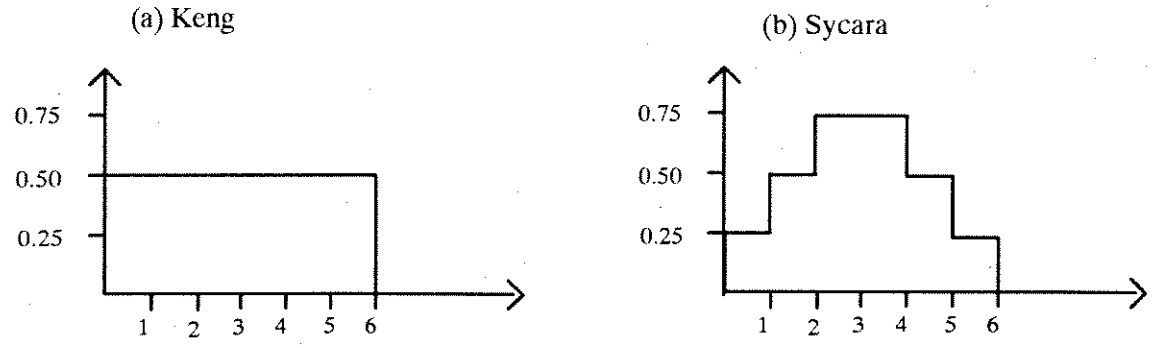


Figura 2.6 - Comparação entre diferentes definições de criticalidades

A proposta de Sycara é desenvolvida para sistemas distribuídos de seqüenciamento, 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 interativo semelhante ao descrito na proposta anterior, utilizando para isso os seguintes passos ;

1. *Seleção da operação com maior criticalidade.* Para cada recurso faz-se a adição das criticalidades, escolhendo-se o recurso com criticalidade total maior e em seguida a operação com maior contribuição na criticalidade total. As duas escolhas são feitas analisando o valor máximo a nível de intervalo unitário de tempo.

2. *Alocação da operação.* Para 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 tempo inicial menos constrangedor" aloca o início no intervalo de tempo com maior "survivability". Os autores afirmam que esta heurística reduz a necessidade de backtracking por infactibilidades que venham a ser detectadas no próximo passo, mas geram alocações pobres em termos de tempo total necessário. Os autores sugerem então, outras heurísticas que podem ser usadas, as quais se baseiam em

outras características importantes do problema, tais como redução nos atrasos, nos estoques intermediários, etc.

3. *Propagação da 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 maiores detalhes. A detecção de infactibilidade implica em "backtracking" para um estado de alocação anterior, modificando-se o instante de início da operação atual.

F. MASCOT - *"Module d'Analyse Sous Contraintes de Problèmes d'Ordonnancement"*.

O sistema MASCOT (Esquirol (1987)) trata do problema de seqüenciamento em jobshops. A abordagem proposta considera como restrições datas limites para a as tarefas utilizando para tal o EBT - "Earliest Beginning Time" e DD - "Due Date", e limitação na oferta de recursos. O sistema MASCOT é na prática uma implementação, utilizando programação lógica, do trabalho de Erschler (1976) de análise sob restrições do problema de seqüenciamento. A modelagem do problema neste sistema é baseado em grafos do tipo potencial-tarefa ("Graphe Potentiel-Tâche" - GPT), sobre os quais é feita uma análise de admissibilidade das seqüências. O sistema utiliza uma base de fatos para representar os dados iniciais do problema e novos fatos gerados durante a sua solução. Estes novos fatos são obtidos a partir de um base de regras que se encarrega de deduzi-los durante a solução do problema. A implementação do sistema utiliza a programação lógica e mais especificamente a linguagem Prolog.

As restrições são classificadas no MASCOT em cumulativas e disjuntivas. Restrições cumulativas resultam da capacidade limitada de recursos e impõem que a cada instante de tempo e para cada recurso a quantidade de recurso utilizada não ultrapasse a quantidade ofertada. As restrições se tornam disjuntivas quando recursos tem capacidade unitária, ou seja, duas tarefas A e B não podem ser processadas simultaneamente (A precede B ou B precede A).

O processo de solução do problema utiliza os seguintes passos :

1 *Inicialização.* Nesta etapa são definidos os conjuntos críticos, os conjuntos disjuntivos. São também determinadas as tarefas iniciais e finais, a partir das datas de início mais cedo - EBT e de entrega - DD. Os conjuntos críticos são definidos a partir das tarefas

cujo consumo acumulado de recursos é maior que a oferta de um determinado recurso. É necessário para resolver este tipo de conflito que existam tarefas que possam ser executadas em intervalos de tempo disjuntos e cujo consumo acumulado de recursos não ultrapasse a oferta. Estas tarefas formam então os chamados conjuntos disjuntos, ou seja, conjuntos cujas tarefas são realizadas em intervalos de tempo distintos.

2. *Atualização das datas limites.* A propagação das datas limites é feita através de dois procedimentos. O primeiro faz a propagação das datas a partir das tarefas iniciais e o segundo a parir das tarefas finais. A propagação das datas limites é feita de forma semelhante a apresentada na seção 2.3.2.A.

3. *Análise disjuntiva.* Nesta fase, são resolvidos os conflitos para subconjuntos de duas tarefas os quais violem restrições temporais e de utilização de recursos. O resultado desta fase é o estabelecimento de relações de precedência temporal entre as tarefas. Utiliza-se aqui um procedimento semelhante ao descrito na seção 2.3.2.C. Caso novas relações sejam obtidas, é necessário atualizar as datas limites encontradas na fase anterior.

4. *Análise cumulativa.* Esta fase se encarrega de resolver os conflitos que não foram totalmente resolvidos na fase anterior e os conflitos associados aos conjuntos críticos de ordem superior a 2, i.e, para conjuntos que tenham mais de duas tarefas gerando um conflito. A exemplo da fase anterior, as alterações relativas as relações de precedência implicam em um retorno a fase de atualização das datas limites.

A figura 2.7 apresenta um diagrama de blocos em grandes linhas do sistema MASCOT.

G. OPAL - "Système d'Ordonnancement Prévisionel d'Atelier"

O OPAL (Bensana (1988) e (1990)), é um sistema para o seqüenciamento preditivo de tarefas para pequenas ou médias empresas. Este sistema é aplicado a jobshops e foi desenvolvido paralelamente ao MASCOT. A função principal deste sistema é a busca de soluções admissíveis. Os objetivos de produção estão expressos em termos de restrições de datas limites (mais cedo e mais tarde), de acordo com o mesmo princípio utilizado no MASCOT, integrando uma versão simplificada do mesmo (Esquirol (1987)).

O OPAL utiliza três fontes de conhecimentos para resolver o problema de seqüenciamento, que são: i) Teórico - obtido da teoria de seqüenciamento (analise de restrições - Erschler (1976)); ii) Empírico - regras de prioridade e suas influências nos

objetivos de produção e iii) Prático - restrições tecnológicas a serem satisfeitas numa dada aplicação. Estas três fontes de conhecimentos são utilizadas pelos seguintes módulos:

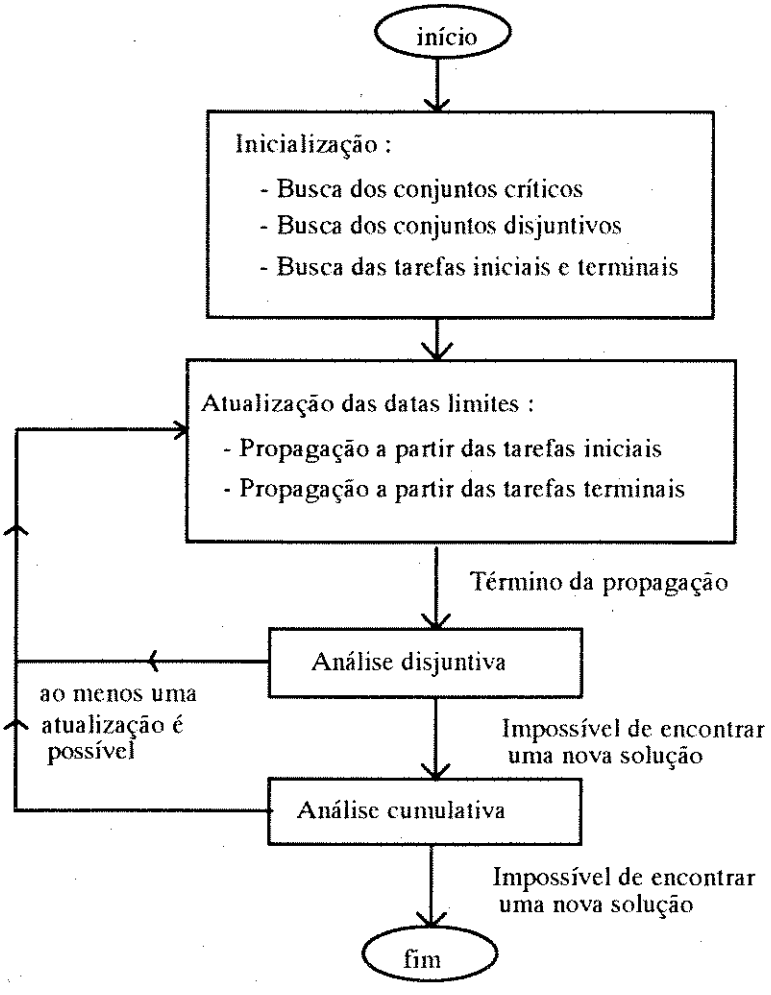


Figura 2.7 : Diagrama de blocos do MASCOT

1. Módulo de análise de restrições - (CBA- "Constraint-based analysis"): avalia as conseqüências das restrições temporais no seqüenciamento das operações, utilizando para tal as condições de Erschler (seção 2.3.2.C). Como resultado deste módulo, podem ser geradas e propagadas novas relações de precedência entre operações.

2. Módulo de tomada de decisão - (DS - "Decision Support Module"): fornece sugestões no seqüenciamento das operações baseado no conhecimento prático ou heurístico. Auxilia o módulo CBA.

3. Supervisor - controla o diálogo entre o CBA e o DS e constrói o "schedule" passo-a-passo de acordo com as decisões tomadas por estes módulos.

O princípio de resolução do OPAL é mostrado na figura 2.8. Quando o CBA não é capaz de seqüenciar as operações, é realizado um retrocesso no algoritmo de busca para encontrar alternativas de possíveis alocações de acordo com sugestões fornecidas pelo módulo de tomada de decisão. Este módulo é responsável então pela solução de conflitos, utilizando várias heurísticas para escolher a melhor solução de acordo com o estado atual da planta. Alguns exemplos de heurísticas utilizadas para resolver os conflitos são:

- seleção do processador com maior número de conflitos;
- seleção de uma operação de particular interesse e
- seleção do processador onde as operações tem a maior folga.

O algoritmo de busca e o mecanismo de retrocesso não são explicitados na referência, o procedimento é heurístico e um retrocesso implica em relaxar as restrições de prazo de entrega das tarefas e em reiniciar a busca da solução.

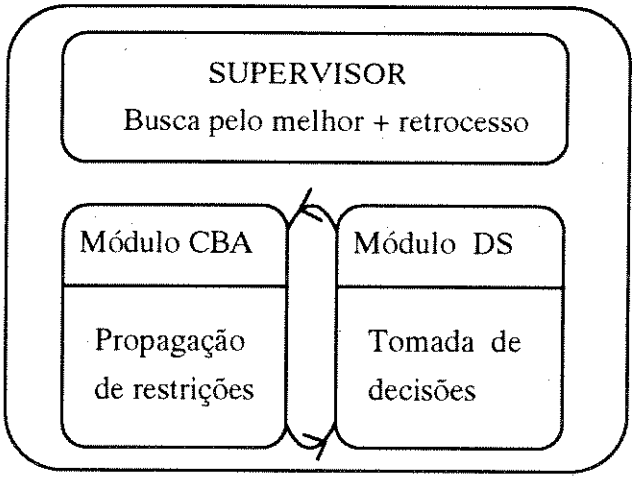


Figura 2.8 : Princípio de resolução do OPAL

H. Comentários gerais e características das propostas

À exceção dos sistema proposto por Egli e Rippin(1988), que trata do problema de seqüenciamento em flowshops, os demais tratam de problemas de jobshops. O enfoque central destes sistemas é resolver conflitos na utilização de recursos de uma forma local, sem garantir que os objetivos de desempenho global da planta sejam atingidos. Estes sistemas apresentam em comum o fato de utilizarem janelas de demanda e técnicas de busca orientadas por restrições.

O sistema ISIS pode ser considerado o precursor na utilização destas técnicas de busca. Neste sistema a busca é direcionada pelas ordens (tarefas), a qual uma vez considerada crítica todas as suas operações também serão. É utilizado no ISIS uma busca do tipo "Beam-search" e uma estrutura multi nível. O sistema OPIS estendeu o sistema ISIS adicionando à perspectiva de ordens utilizada no mesmo, uma nova perspectiva de recursos, tornando o sistema mais eficiente. Já o CORTES utiliza uma abordagem micro-oportunísta, a qual considera operações em vez de tarefas.

Outros sistema que utilizam esta última abordagem baseada nas operações são os sistemas ISPS (Keng (1988)) e DCHS (Sycara (1991)), os quais utilizam o conceito de criticalidade das operações. Isto é feito nos dois sistemas de forma bastante semelhante, utilizando um algoritmo essencialmente heurístico. A principal diferença entre eles, é que o sistema DCHS trata de problemas distribuídos, utilizando uma abordagem multi-agente.

Os sistemas MASCOT e OPAL utilizam uma abordagem baseada na admissibilidade das seqüências, utilizando para tal uma técnica de raciocínio temporal segundo o princípio de propagação de restrições temporais proposto por Erschler (1976). Estes sistemas são aplicados ao problema de seqüenciamento de jobshops e à gestão de projetos. Uma extensão à estes sistemas é o MASCOT2 (Lopez (1991)) que utiliza, além do raciocínio temporal, um raciocínio "energético", onde o problema é modelado por intervalos de tempo associados às tarefas e aos recursos, tratados como consumidores (tarefas) e fornecedores (recursos) de uma certa quantidade de energia expressa em [recursos x tempo] como por exemplo: homens-dia, kilowatts-hora, etc

O interesse nestas abordagens está nos conceitos ou técnicas utilizadas, que embora não possam ser extrapolados diretamente para o problema do flowshop tratado neste trabalho, auxiliaram na elaboração da proposta. Estes conceitos ou técnicas auxiliam na redução da árvore de busca e são dos seguintes tipos:

- propagação de restrições temporais;
- janelas de tempo associadas às tarefas e operações;
- medidas de criticalidade;
- procedimentos heurísticos ("Beam-search" por exemplo); etc.

À seguir serão apresentados dois trabalhos que tratam do problema de flowshops na Indústria de Processos Químicos Descontínuos. Rodrigues (1992), seção 2.4.2, propõe em seu trabalho algoritmos para resolver o problema de seqüenciamento ótimo de flowshops com restrições na oferta de recursos de uso compartilhado na Indústria de Processos Químicos. Os conceitos de base deste trabalho foram obtidos a partir do

trabalho de Rodrigues (1992). Outro trabalho que também utiliza os conceitos acima, é o trabalho de Campos (1993), seção 2.4.3, o qual incorpora aos algoritmos propostos por Rodrigues a existência de tarefas que são alocadas externamente a partir da definição de janelas de tempo formadas pelas datas de início mais cedo e de término mais tarde das tarefas.

2.4.2 Branch and Bound com busca orientada pelas restrições sobre recursos compartilhados.

Em Rodrigues (1992) é proposto um algoritmo "*Branch and Bound - BAB*" que utiliza as restrições sobre os recursos compartilhados para o cálculo do limitante inferior ("*lower bound - lb*") em cada nó da árvore da busca.

Para tanto o problema é definido como segue:

- cada nó é caracterizado por um conjunto de tarefas alocadas K-1, um conjunto de tarefas ainda não alocadas U+1 e um limitante inferior;
- as operações de cada tarefa alocada em cada nó expandido são alocadas satisfazendo as restrições sobre os recursos;
- a disponibilidade dos recursos (r=1,2,...R) é dada por $q^r(t)$ [unidades recurso/unidade tempo];
- a demanda do recurso r pela tarefa i no processador j é dada por $Q_{ij}^r(t)$ [u.r/u.t]
- o tempo de processamento da tarefa i no processador j é dado por t_{ij} [u.t].

Em cada nó gerado, que corresponde a uma seqüência parcial K, o lb será obtido como a soma de duas parcelas, uma associada com a seqüência K que é facilmente obtida e outra que deve ser uma estimativa do tempo necessário para realizar as tarefas U ainda não alocadas.

No nó raiz o limitante inferior é definido como o tempo mínimo necessário para processar todas as tarefas com a hipótese de utilização total dos recursos compartilhados, ou seja T tal que:

$$T = \max_{1 \le r \le R}[T^r] \tag{2.6}$$

onde: T^r é dado por: $\int_{t=0}^{t=T^r} q^r(t)dt = \sum_{i=1}^{N}\sum_{j=1}^{M}\int_{t=0}^{t=t_{ij}} Q_{ij}^r(t)dt = D^r$ (2.7)

N = quantidade total de tarefas

M = quantidade total de processadores

D^r = a demanda total pelo recurso r.

Nos nós intermediários a contribuição das tarefas ainda não alocadas é estimada utilizando o conceito das Regiões I, II e III apresentado na seção 2.3.3.

No algoritmo A propõe-se estimar o limitante inferior da seqüência parcial K definindo a duração da região III a partir do tempo necessário para obter a quantidade de recurso não utilizado em função do incremento da região I. Este cálculo deve ser feito para cada recurso e o limitante inferior será definido pelo maior aumento de tempo devido à quantidade de recurso não utilizada na região I. A figura 2.9 ilustra esta proposta.

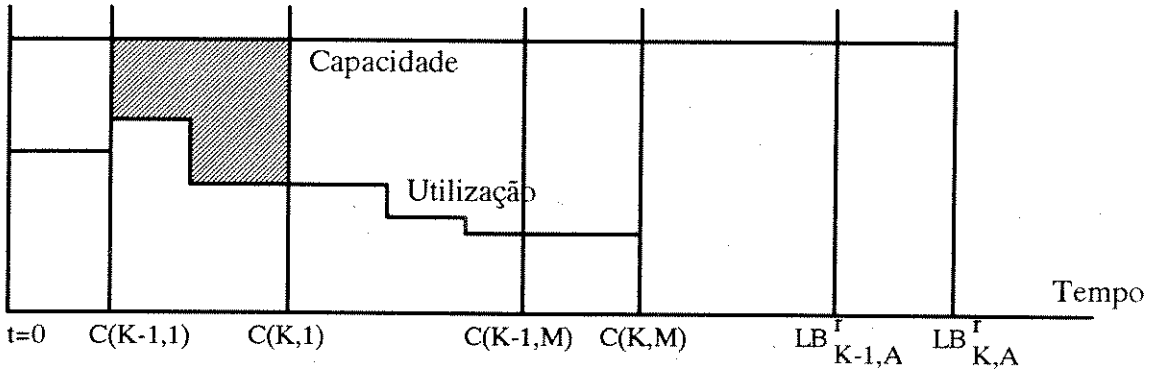


Figura 2.9 : Algoritmo A: Recurso não utilizado na região I e aumento do "lower bound"

É feita a hipótese de que nas Regiões II e III os recursos compartilhados serão utilizados completamente. O recurso r não utilizado completamente entre C(K-1,1) e C(K,1), quando a nova tarefa é introduzida terá como consequência um aumento no limitante inferior da seqüência K-1, LB^r_{K-1}, levando a um limitante inferior para a seqüência K, LB^r_K, dado pela seguinte equação :

$$\int_{t=LB^r_{K-1,A}}^{t=LB^r_{K,A}} q^r(t)dt = \int_{t=C(K-1,1)}^{t=C(K,1)} \{q^r(t) - \sum_{i\in K}\sum_{j=1}^{M} Q^r_{ij}(t)Z_{ij}(t)\}dt \qquad (2.8)$$

onde:

$$Z_{ij}(t) = \begin{cases} 1 \text{ se a tarefa i está no processador j no instante t} \\ 0 \text{ nos outros casos} \end{cases}$$

O limitante inferior pode também ser escrito como o limite da Região III na seqüência parcial K. Da equação anterior e utilizando a equação 2.9 que representa a hipótese de utilização completa dos recursos nas regiões II e III pela seqüência K, obtém-se a equação 2.10 que define a duração da Região III.

$$D^r = \sum_{i \in K} \sum_{j=1}^{M} \int_{t=0}^{t=C(K-1,1)} Q_{ij}^r(t) Z_{ij}(t) dt + \int_{t=C(K-1,1)}^{t=C(K-1,M)} q^r(t) dt + \int_{t=C(K-1,M)}^{t=LB^r_{K-1,A}} q^r(t) dt \qquad (2.9)$$

$$\int_{t=C(K,M)}^{t=LB^r_{K,A}} q^r(t) dt = D^r - \sum_{i \in K} \sum_{j=1}^{M} \int_{t=0}^{t=C(K,1)} Q_{ij}^r(t) Z_{ij}(t) dt - \int_{t=C(K,1)}^{t=C(K,M)} q^r(t) dt \qquad (2.10)$$

O limitante inferior para a seqüência parcial K é então dado pela equação 2.11.

$$LB_{K,A} = \max_{1 \le r \le R} \{LB^r_{K,A}\} \qquad (2.11)$$

O algoritmo A pode ser melhorado levando em consideração que o início da primeira operação da próxima tarefa a ser alocada (levando à seqüência parcial K+1) pode não ser C(K,1). Isto será devido a que o recurso já utilizado na Região II pode deixar um saldo insuficiente para esta operação, obrigando que esta operação tenha que ser atrasada. Esta situação está ilustrada na figura 2.10.

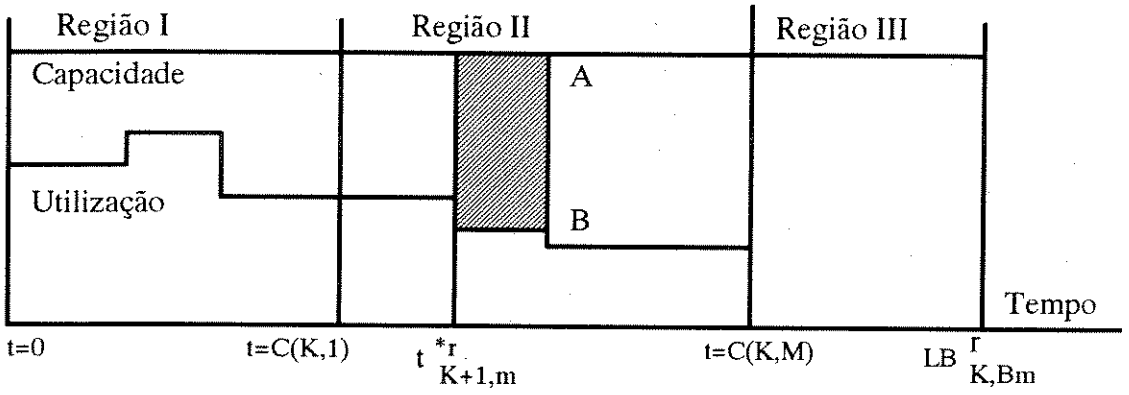


Figura 2.10. A primeira operação da tarefa candidata m para a seqüência K+1 pode ser iniciada somente em $t^{*r}_{K+1,m}$ devido ao consumo de recurso (AB) e a disponibilidade.

A região marcada representa o consumo de recurso pela primeira operação da tarefa candidata m a ser acrescida à seqüência parcial K para gerar a seqüência parcial K+1. O perfil de demanda desta operação (um valor AB constante na figura) não permite

iniciar esta operação em C(K,1), mas apenas em $t^{*r}_{K+1,m}$. Quando isto acontece o recurso não utilizado entre C(K,1) e $t^{*r}_{K+1,m}$ será perdido.

No algoritmo B esta informação é utilizada para obter uma melhor estimativa do limitante inferior de uma forma semelhante à utilizada no algoritmo A. O processo é mais complexo porque em cada nó deve se proceder a: i) obter $t^{*r}_{K+1,m}$ para a primeira operação de cada tarefa candidata e para cada recurso r = 1,..,R; ii) deve se determinar o recurso que leva ao valor máximo; e iii) a tarefa que leva ao menor aumento será aquela que define o limitante inferior (de forma a preservar a característica monotônica do limitante inferior).

Para o algoritmo B as equações anteriores são modificadas introduzindo o consumo do recurso no intervalo (C(K,1), $t^{*r}_{K+1,m}$) e modificando os limites de integração como mostrado nas equações 2.12 e 2.13.

$$\int_{t=LB^r_{K-1,B}}^{t=LB^r_{K,Bm}} q^r(t)dt = \int_{t=C(K-1,1)}^{t=t^{*r}_{K+1,m}} \{q^r(t) - \sum_{i\in K}\sum_{j=1}^{M} Q^r_{ij}(t)Z_{ij}(t)\}dt \qquad (2.12)$$

$$\int_{t=C(K,M)}^{t=LB^r_{K,Bm}} q^r(t)dt = D^r - \sum_{i\in K}\sum_{j=1}^{M} \int_{t=0}^{t=t^{*r}_{K+1,m}} Q^r_{ij}(t)Z_{ij}(t)dt - \int_{t=t^{*r}_{K+1,m}}^{t=C(K,M)} q^r(t)dt \qquad (2.13)$$

O limitante inferior da seqüência parcial K seguida da tarefa m é dado pelo valor máximo de $LB^r_{K,Bm}$ quando se consideram todos os recursos:

$$LB_{K,Bm} = \max_{1\le r\le R}\{LB^r_{Bm}\} \qquad (2.14)$$

Finalmente o limitante inferior para a seqüência K é dado pela equação 2.15 como o valor mínimo que se obtém quando se consideram todas as tarefas candidatas:

$$LB_{K,B} = \min_{m\in U}\{LB_{K,Bm}\} \qquad (2.15)$$

Comentários sobre os algoritmos A e B.

• No cálculo do limitante inferior da seqüência parcial K os dois algoritmos fazem a alocação das operações da última tarefa *antes* do cálculo. Desta forma este leva em

consideração o aumento em C(K,M) que pode resultar como consequência da necessidade de atrasos nas operações da última tarefa devidos às restrições sobre os recursos. Esta alocação compatível com os recursos disponíveis tem que ser feita de qualquer forma se o algoritmo BAB tem como objetivo analisar soluções factíveis durante o processo de busca. Uma factibilização realizada apenas a-posteriori sobre a solução final não é adequada porque o aumento necessário do "makespan" pode ser muito grande (Rodrigues (1992)).

• Em relação ao algoritmo A, o algoritmo B realiza uma espécie de previsão do aumento do limitante inferior. Se todas as possíveis tarefas candidatas a serem acrescidas à seqüência parcial K tem que se atrasadas devido a problemas com recursos, o atraso mínimo representa uma perda de recursos que pode ser refletida no limitante inferior. Esta previsão representa um cálculo a mais, especificamente a obtenção de $t^{*_J}_{K+1,m}$, mas estes valores terão que ser calculados de qualquer forma se o nó da árvore correspondente à seqüência parcial {K,m} é visitado.

• As equações que descrevem os algoritmos A e B são fáceis de implementar se o consumo de cada recurso é considerado constante durante o tempo de processamento de uma operação. Esta hipótese pode ser freqüentemente aceita *per se* ou pode corresponder a um consumo médio utilizado num modelo agregado da planta utilizado numa estrutura de seqüenciamento a dois níveis como descrito no capítulo 3.

• Os dois algoritmos não levam em consideração que os "lower bounds" podem estar principalmente definidos pelas relações de precedência em situações em que a oferta de recursos não é crítica. A inclusão de uma estimativa dos "lower bounds" através do SMBB ("*Single Machine Based Bound*") ou FMBB ("*Full Machine Based Bound*") é proposta nos algoritmos C_1 e C_2.

Algoritmos C_1 e C_2.

Os algoritmos A e B utilizam uma perspectiva baseada nos recursos para o cálculo dos "lower bounds". As relações de precedência, traduzidas num tempo mínimo de processamento, também contribuem à definição do limitante inferior do "makespan" e serão importantes se as restrições sobre os recursos compartilhados não são muito fortes. Com o objetivo de combinar as duas perspectivas são propostos os algoritmos C_1 e C_2, como modificações dos algoritmos A e B, nos quais o limitante inferior é calculado como o valor máximo entre o valor obtido por eles e o valor estimado a partir da heurística FMBB.

Heurística FMBB - "Full Machine Based Bound"

A seguir são apresentadas as equações para o cálculo do valor do limitante inferior pela heurística FMBB, adaptada à utilização das regiões I, II e III definidas acima.

$$B_j = C(K,j) + \sum_{i \in J} t_{ij} + \min_{i \in J}\left(\sum_{l=j+1}^{M} t_{il} \right) \qquad 1 \le j \le M-1 \tag{2.16}$$

$$B_M = C(K,M) + \sum_{i \in J} t_{iM} \tag{2.17}$$

$$LB_{FMBB} = \max_{i \le j \le M} \{B_j\} \tag{2.18}$$

2.4.3 Branch and Bound com busca orientada pelas restrições sobre prazos e recursos compartilhados

Em Campos (1993) é proposta uma extensão aos algoritmos propostos por Rodrigues (1992) e apresentados na seção 2.4.2 os quais tratam do problema do flowshop na Indústria de Processos Químicos. Neste algoritmo Campos considera além das restrições sobre recursos de uso compartilhados (utilidades), também restrições sobre os prazos de entrega das tarefas (produtos).

O problema é definido da seguinte forma: são dadas N tarefas, cada uma das quais podendo ter um tempo de pronto ("ready time" - RT) e um prazo de entrega ("due date"- DD) especificado pelo planejador, M processadores, R recursos compartilhados com disponibilidade instantânea limitada e armazenagem intermediária ilimitada. O objetivo é seqüenciar as tarefas de forma a minimizar o "makespan" considerando seqüências de permutação.

A atribuição de um tempo de pronto e de um prazo de entrega para uma tarefa é denominada *alocação externa* e define uma *janela de demanda* para as tarefas. As tarefas alocadas externamente são denominadas *tarefas externas* e as demais *tarefas internas*. Estas alocações externas podem ocorrer em função de restrições tecnológicas, manutenção preventiva, atendimento a clientes preferenciais, etc. É possível também definir uma janela de demanda para as tarefas internas considerando para tal RT = 0 e DD = TF, onde TF é uma estimativa do tempo máximo para processar todas as tarefas.

Geralmente as restrições sobre o prazo de entrega e sobre a oferta de recursos compartilhados são considerados na literatura como restrições flexíveis, as quais podem

ser relaxadas mediante um certo custo, o que possibilita simplificar o problema. Neste trabalho considera-se que estas restrições não podem ser relaxadas sob pena de se chegar a uma solução final infactível ou pobre do ponto de vista econômico.

A solução do problema de flowshop proposto por Rodrigues (1992) consiste na determinação do momento exato em que as operações de cada tarefa devem ser realizadas de forma a minimizar o "makespan", maximizando a utilização dos recursos de uso compartilhado. Já Campos, por considerar alocações externas de tarefas, adiciona duas novas decisões ao problema:

1) determinar quando as tarefas com RT $\neq$ 0 devem fazer parte do processo de solução do problema e

2) determinar quando a alocação de uma tarefa provoca a violação do DD de outra tarefa, já que existem tarefas com prazos de entrega bem definidos.

O problema é então resolvido utilizando duas técnicas:

1) um algoritmo de busca do tipo "Branch and Bound" - BAB, que permite obter a solução ótima do problema calculando o limitante inferior de cada nó com base na oferta de recursos compartilhados, utilizando os algoritmos propostos por Rodrigues. Utiliza-se no BAB um algoritmo de busca em profundidade, o qual permite determinar rapidamente um limitante superior ("upper bound"), que serve para eliminar ramos na árvore de decisão e definir DD para as tarefas internas; e

2) um algoritmo de busca do tipo "Constraint Heuristic Search", que busca as soluções do problema que satisfazem todas as restrições do problema, eliminando antecipadamente da árvore de busca todos os ramos que levam a soluções infactíveis.

O algoritmo, proposto por Campos (1993), consiste de um pré-processamento que é aplicado a cada nó da árvore de busca, antes de sua explosão, para selecionar, dentre todas as candidatas a fazer parte da seqüência parcial daquele nó, aquelas que potencialmente levarão a uma seqüência final factível. A determinação do conjunto de tarefas candidatas, denominado **CAN**, é feito através da análise de restrições temporais originadas: i) pelos limites de tempo das tarefas; e ii) pela disponibilidade de recursos.

A determinação do conjunto **CAN** é feito primeiro determinando um conjunto inicial de tarefas candidatas e posteriormente eliminando deste conjunto aquelas tarefas que podem resultar em soluções infactíveis. As tarefas que farão parte inicialmente do conjunto **CAN** são todas aquelas que possuem seu RT dentro do intervalo [C(K,1), C(K,M)], ou seja, na região II conforme definido na seção 2.3.3, onde K é a seqüência parcial correspondente ao nó que está sendo pesquisado. Na hipótese de todas as tarefas

pertencentes ao conjunto **CAN** inicial infactibilizarem a alocação de algumas tarefas com RT maior do que o C(K,M), estas últimas são adicionadas ao conjunto **CAN**, aquelas que provocaram a infactibilização são eliminadas através de um mecanismo de propagação de restrições. A eliminação de tarefas deste conjunto é feita por três procedimentos: 1) análise das restrições temporais do problema; 2) utilizando uma regra heurística e 3) pela determinação de infactibilidades resultantes da disponibilidade limitada de recursos de uso compartilhados.

O procedimento de análise de restrições está baseado nas condições de factibilidade de uma solução proposta por Erschler (1986) e apresentadas na seção 2.3.2.C. Dadas duas operações, (i,m) e (j,m), que podem ser operações reais ou fictícias (estas últimas resultantes da união de um conjunto de operações), a análise de restrições procura determinar se os limites de tempo destas operações implicam em algum ordenamento entra elas. A figura 2.11 mostra as situações possíveis para um flowshop . Nesta figura, T_{ij} é o intervalo de tempo disponível para processar as operações (i,m) e (j,m) supondo que i será processada antes de j, T_{ji} é definido de forma análoga e TP é o tempo necessário para processar estas duas operações.

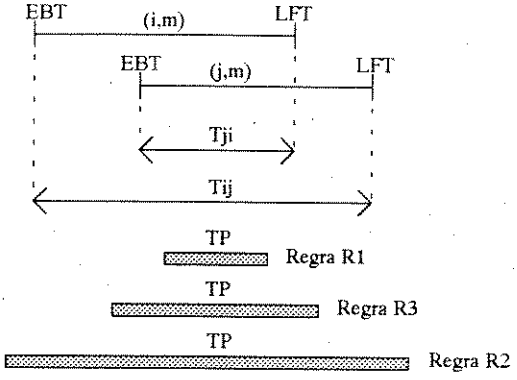


Figura 2.11 : Análise das restrições temporais do problema

A partir destas definições é possível estabelecer três regras:

Regra R1 : **Se** TP ≤ T_{ij} e TP ≤ T_{ji} **então** não é possível estabelecer a priori um ordenamento entre i e j;

Regra R2 : **Se** TP ≥ T_{ij} e TP ≥ T_{ji} **então** não existe uma solução que satisfaça as restrições temporais das operações e portanto o nó que está sendo explorado é infactível e deve ser eliminado da árvore de busca;

Regra R3 : **Se** TP ≤ T_{ij} e TP ≥ T_{ji} **então** existem três possibilidades:

1) **Se** i e j são tarefas reais **então** j deve ser eliminada de **CAN**;

2) **Se** i é uma tarefa fictícia e j uma tarefa real **então** j deve ser eliminada de **CAN**; e

3) **Se** i é uma tarefa real e j uma tarefa fictícia **então** propõe-se a utilização de regras heurísticas para eliminar uma ou mais tarefas candidatas do conjunto **CAN**.

A regra heurística é utilizada quando a Regra R3.3 é encontrada. A estratégia adotada é a de atrasar a tarefa menos crítica, sendo que a criticalidade de uma tarefa é calculada pela equação 2.19. A criticalidade calculada desta forma generaliza o conceito de criticalidade apresentadas por Keng e Sycara nas seções anteriores, pois ela leva em consideração para o cálculo da criticalidade além do tempo disponível para processar a tarefa, a quantidade de recurso envolvida. Desta forma, para duas operações com a mesma relação (TP/DW), será considerada mais crítica aquela que necessitar de mais recursos para a sua realização. Da mesma forma, entra duas tarefas com a mesma relação (Q_{ij}/q) será mais crítica aquela que tiver menor janela de tempo disponível para a alocação.

$$crt(i) = \sum_{i \le j \le M} \left(\frac{TP_{ij}}{DW_{ij}} \times \max_{1 \le r \le R} \left(\frac{Q_{ij}^{r}}{q^{r}} \right) \right) \tag{2.19}$$

onde:

TP_{ij} = tempo de processamento da operação i da tarefa j

DW_{ij} = intervalo de tempo disponível para processar a operação i da tarefa j

Q_{ij}^{r}= consumo instantâneo do recurso r pela operação i da tarefa j

q^{r} = disponibilidade instantânea do recurso r

A determinação de infactibilidades resultantes da disponibilidade limitada de recursos de uso compartilhados é realizada após a análise das restrições temporais. Este procedimento consiste na eliminação de eventuais tarefas pertencentes ao conjunto **CAN** cuja alocação infactibilize alguma tarefa ainda não seqüenciada, ou seja, é um procedimento de verificação de infactibilidade devido às restrições associadas aos recursos e não ao tempo.

O algoritmo utiliza uma busca em profundidade o qual possibilita uma busca mais eficiente no espaço de solução do problema, através da rápida obtenção de um "upper bound". A utilização da heurística como proposta por Campos reduz significativamente o espaço de busca, sem perder a solução ótima na grande maioria dos casos, como verificado experimentalmente por ele.

2.5 Conclusão

Neste capítulo foram apresentados os conceitos de base utilizados neste trabalho. O problema de seqüenciamento é enfocado sob o contexto preditivo, utilizando como técnica de solução uma abordagem multi nível em conjunto com um algoritmo de busca em árvore do tipo "Branch And Bound" com objetivo de encontrar soluções ótimas.

Os conceitos e técnicas apresentados neste capítulo e incorporadas ao algoritmo proposto foram de fundamental importância para a formulação da proposta. Dentre eles estão: o conceito de janela, medidas de criticalidades, análise e propagação de restrições, agregação de dados e estruturas hierárquicas.

A complexidade do problema de flowshop tal qual ele é tratado neste trabalho, tem a sua formulação matemática bastante facilitada pelo tipo de abordagem utilizada. Isto se deve ao fato da utilização da abordagem multi nível a qual possibilita decompor o problema e resolvê-lo hierarquicamente através de diferentes níveis de abstração.

Capítulo 3 - Algoritmo de seqüenciamento multi nível

3.1 Introdução

Neste capítulo estuda-se o problema de seqüenciamento ("scheduling") em flowshops com restrições na oferta de recursos compartilhados e com datas de início e de término das tarefas definidas externamente. A técnica proposta para resolver este problema é do tipo busca orientada pelas restrições e utiliza uma estrutura multi nível onde o processo de busca é realizado sobre um modelo do problema agregado e simplificado. A simplificação do problema é conseguida através de um pré-processamento que determina os recursos mais críticos do ponto de vista do impacto sobre o "makespan". A técnica proposta está baseada em:

i) um processo heurístico para a classificação dos recursos baseado na demanda total de cada recurso, no consumo acumulado instantâneo devido à coexistência de operações na planta, e no conceito de criticalidade das operações, sendo este último apresentado por Keng (1988) e Sycara (1991);

ii) um algoritmo de busca do tipo BAB, com limitante inferior calculado com base na utilização dos recursos de uso compartilhado, proposto originalmente por Rodrigues (1992) e adaptado por Campos (1993) para a consideração de tarefas com datas de término definidas externamente trabalhando sobre um modelo agregado e simplificado; e

iii) uma fase de factibilização onde se utiliza o modelo completo da planta, considerando as operações de preparação, processamento e transferência das tarefas individualmente.

Estas técnicas são utilizadas em conjunto com o objetivo de obter soluções ótimizadas para problemas de flowshop mais reais do que os tratados geralmente na literatura de pesquisa operacional.

O capítulo está estruturado da seguinte forma:

- na seção 3.2 é apresentada uma descrição simplificada do problema;
- na seção 3.3 é apresentada a estrutura geral do algoritmo multi nível, discutindo a agregação de dados, as diferentes propostas para classificação dos recursos, o processo de busca e a análise de factibilidade; e
- na seção 3.4 é feita uma descrição global do algoritmo visando definir as interfaces existentes entre as diversas funções e o planejador.

3.2 Descrição do problema

O problema é definido da seguinte forma: são dadas N tarefas, das quais J ($J \leq N$) foram alocadas externamente, M processadores, R recursos compartilhados, armazenagem intermediária ilimitada. O objetivo é o de seqüenciar as tarefas de forma a minimizar o "makespan", respeitando a oferta limitada de recursos e as datas de término definidas externamente. A solução é restrita a ser uma seqüência de permutação e não é permitida a preempção das operações.

Uma alocação externa consiste na atribuição de um tempo de pronto ("Ready Time"-RT) e de um prazo de entrega ("Due Date"-DD) para uma tarefa, ou seja, na atribuição de uma janela de tempo "conveniente" para ela. Esta janela deverá sempre ser maior ou igual ao somatório dos tempos de processamento das operações da tarefa. Sendo igual a alocação é denominada fixa, caso contrário ela é denominada flexível. As tarefas alocadas externamente são denominadas tarefas externas, e as demais denominadas tarefas internas.

A produção na Indústria de Processos Químicos - IPQ é marcada pela existência de utilidades do tipo vapor, energia elétrica, etc, as quais devido às limitações na quantidade ofertada restringem a coexistência de operações em diferentes processadores na planta, o que faz com que seja de fundamental importância a consideração destas utilidades na solução do problema de seqüenciamento (Rodrigues (1992)). Isto exige que o mesmo não se limite ao ordenamento das tarefas, como nos problemas de flowshop de um modo geral, mas que seja também necessária a programação horária das diversas operações existentes.

A figura 3.1 mostra uma representação esquemática do processo de fabricação em um ambiente do tipo flowshop, composto de M processadores, armazenagem intermediária e diversas utilidades.

3.3 Algoritmo multi nível: Estrutura e elementos

Para resolver o problema de seqüenciamento é proposta uma abordagem multi nível, a qual consiste de três níveis distintos : pré-seqüenciamento, seqüenciamento e pós-seqüenciamento. A figura 3.2 mostra a hierarquia existente entre os três níveis. O nível de pré-seqüenciamento tem como objetivo a determinação de um modelo agregado e simplificado da planta e das tarefas, de modo que o processo de busca orientada pelas restrições utilizado na fase de seqüenciamento seja mais eficiente. É bem conhecido que os

processos de busca implicam em um esforço computacional grande e, portanto, o que se pretende é em vez de utilizar menos tempo em todas as decisões, utilizar um tempo maior nas decisões importantes.

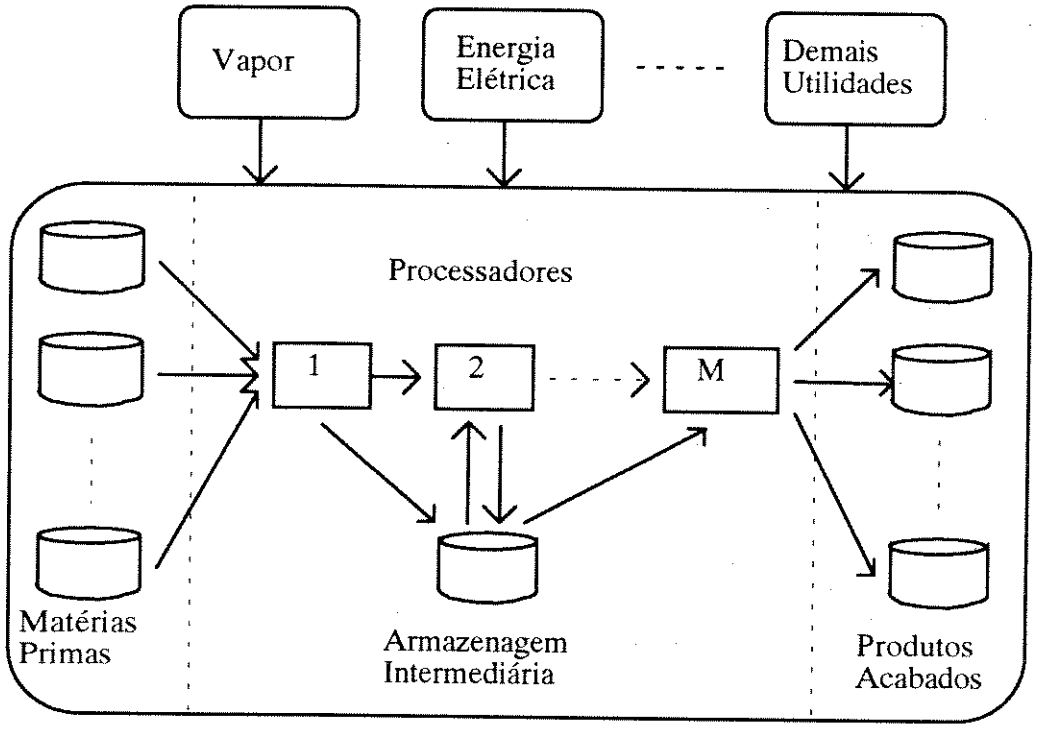


Figura 3.1 : Representação esquemática do processo

No nível de seqüenciamento o problema é decomposto em dois níveis (ou fases): a fase de geração de seqüências, onde é utilizado um algoritmo de BAB e a fase de factibilização onde é realizada uma simulação do lançamento das tarefas na planta para verificação da factibilidade. Ocorrendo alguma infactibilidade tem-se um retrocesso ("backtracking") ao nível de geração de seqüências.

No nível de pós-seqüenciamento é realizada uma análise detalhada da utilização de todos os recursos da seqüência final gerada no nível anterior com o objetivo de gerar a carta de Gantt final, e fornecer os gráficos detalhados da utilização dos recursos. Neste nível são também fornecidas as seqüências alternativas geradas durante a solução do problema, para que possam eventualmente serem utilizadas caso a primeira se torne infactível por algum dado novo.

A seguir cada um dos pontos mencionados acima será discutido detalhadamente.

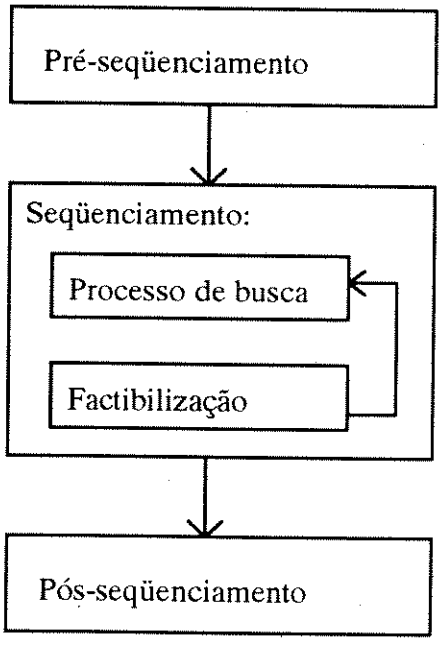


Figura 3.2 : Níveis de Seqüenciamento

3.3.1 Pré-seqüenciamento: Agregação de dados

A agregação dos dados tem por objetivo possibilitar a criação de um modelo de dados agregados, de tal forma a se obter dois diferentes níveis de abstração a serem tratados de forma seletiva pela abordagem hierárquica existente no nível de seqüenciamento.

São realizados dois tipos de agregação diferentes: a agregação dos tempos de preparação, processamento e transferência, e a agregação das quantidades de recursos necessárias à execução de cada operação.

A agregação dos tempos de preparação, processamento e transferência tem por finalidade permitir que estas operações possam ser tratadas como um único bloco e representados por uma variável agregada.

No que se refere ao consumo dos recursos, existem dois tipos de agregações. O primeiro tipo está relacionado com a simplificação da manipulação da função que representa os consumos das operações e da oferta dos recursos, e o segundo tipo à agregação dos tempos das operações.

A. Agregação dos consumos

O perfil de consumo das operações é representado por uma função do tempo. Uma forma de simplificar esse perfil é através da média de consumo de tal forma a tornar este perfil constante e representá-lo através de um valor agregado (constante). Este procedimento torna os cálculos a serem realizados na fase de seqüenciamento mais simples pela manipulação de um número menor de variáveis.

A agregação dos valores que é realizada preserva a quantidade total de recursos demandada pelas operações, de tal forma que esta quantidade seja a mesma em todos os níveis do seqüenciamento. O inconveniente é que caso ocorram grandes desvios nestes valores, o valor médio pode não ser muito significativo em termos da análise da quantidade instantânea demandada. Entretanto, durante a fase de factibilização a análise detalhada levará em consideração a demanda real.

A média de consumo para a operação de processamento pode ser obtida através da seguinte equação :

$$Qp_{ij}^{r} = \frac{\int_{t=0}^{t=tp_{ij}} qp_{ij}^{r}(t)dt}{tp_{ij}} \qquad (3.1)$$

onde :

Qp_{ij}^{r} = variável agregada que representa o perfil de consumo de uma operação de processamento

$qp_{ij}^{r}(t)$ = quantidade de recurso r necessária para o processamento da operação j da tarefa i no instante t

tp_{ij} = tempo de processamento da operação j da tarefa i

Este tipo de agregação é também realizada para os perfis associados às operações de preparação e de transferência das tarefas. Tem-se :

$$Qs_{ij}^{r} = \frac{\int_{t=0}^{t=ts_{ij}} qs_{ij}^{r}(t)dt}{ts_{ij}} \qquad (3.2)$$

onde :

Qs_{ij}^{r} = variável agregada que representa a média do consumo de uma operação de preparação

$qs_{ij}^{r}(t)$ = quantidade de recurso r necessária para a preparação da operação j da tarefa i no instante t

ts_{ij} = tempo de preparação da operação j da tarefa i, e

$$Qt_{ij}^{r} = \frac{\int_{t=0}^{t=tt_{ij}} qt_{ij}^{r}(t)dt}{tt_{ij}} \tag{3.3}$$

onde :

Qt_{ij}^{r} = variável agregada que representa a média do consumo de uma operação de transferência

$qt_{ij}^{r}(t)$ = quantidade de recurso r necessária para a transferência da operação j para a operação j+1 da tarefa i no instante t

tt_{ij} = tempo de transferência da operação j da tarefa i

Estas três variáveis agregadas serão utilizadas para definir o consumo agregado total da operação.

O perfil de oferta dos recursos é tratado de forma semelhante aos perfis de consumo dos recursos, e a obtenção do valor agregado é feito da seguinte forma:

$$q^{r} = \frac{\int_{t=0}^{t=T} q^{r}(t)dt}{T} \tag{3.4}$$

onde :

q^{r} = variável agregada que representa a oferta de recursos

$q^{r}(t)$ = oferta do recurso r no instante t

T = Tempo máximo necessário para a execução de todas as tarefas, onde:

$$T = \sum_{i=1}^{N}\sum_{j=1}^{M} t_{ij} \tag{3.5}$$

B. Agregação dos tempos

A agregação dos tempos relativos às operações de preparação, processamento e transferência está diretamente associada à forma pela qual estas operações são realizadas na planta. O processamento da operação j da tarefa/produto i, que no flowshop é também realizada no processador j, é realizada da seguinte forma:

1) prepara o processador j para processar o produto i;

2) transfere o produto i do processador j-1 para o processador j;

3) processa o produto i;

4) transfere o produto i para o processador j+1 ou para a armazenagem caso este esteja ocupado; e

5) prepara o processador j para processar o produto i+1.

A figura 3.3 mostra de forma esquemática a subdivisão de uma operação, e a simultaneidade de ocupação de dois processadores durante a transferência de produtos entre dois processadores adjacentes.

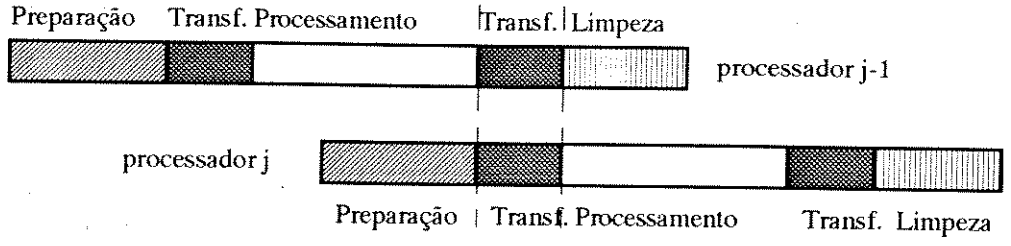


Figura 3.3 : Subdivisão de operações e transferência de produtos entre o processador j e j-1

Os passos 1 e 5, apesar de serem ambos operações de preparação, se referem respectivamente à troca de dispositivos ("setup") e a limpeza e eventual descontaminação dos processadores. Uma simplificação utilizada é a de se assumir que os passos 1 e 5 são realizados simultaneamente. Neste caso temos o seguinte:

1) transferência do produto i do processador j-1 para o processador j;

2) processamento do produto i;

3) transferência do produto i para o processador j+1 ou para a armazenagem; e,

4) preparação do processador j para processar o produto i+1.

A agregação do tempo é realizada então somando-se os tempos de preparação, processamento e transferência dos produtos a serem seqüenciados de acordo com os passos definidos acima. Esta agregação dos tempos, faz com que os perfis de utilização dos recursos das operações associadas a estes tempos devam ser também agregados. A forma proposta para esta agregação, é a de considerar para todo o intervalo, i.e., o tempo agregado, um valor constante de consumo obtido a partir da soma ponderada do consumo das operações de preparação, processamento e transferência. Desta forma, o tempo agregado das operações, considerando que os tempos de transferência de e para o processador são iguais, é dado por :

$$t_{ij} = ts_{ij} + tp_{ij} + 2 \times tt_{ij} \qquad (3.6)$$

a quantidade de recurso agregada para a operação j da tarefa i, é dada por :

$$Q_{ij}^{r} = \frac{(ts_{ij} \times Qs_{ij}^{r}) + (tp_{ij} \times Qp_{ij}^{r}) + 2 \times (tt_{ij} \times Qt_{ij}^{r})}{ts_{ij} + tp_{ij} + 2 \times tt_{ij}} \tag{3.7}$$

onde :

Q_{ij}^{r} = quantidade agregada de recursos necessários para a realização da operação j da tarefa i

3.3.2 Pré-seqüenciamento: Classificação dos Recursos

A classificação dos recursos tem como objetivo definir a importância relativa dos mesmos no processo de obtenção da solução do problema. Em qualquer problema real o planejador sabe que certos recursos são críticos e que as decisões de alocação das operações são tomadas preferencialmente em função dos mesmos.

O objetivo é o de analisar o caráter crítico de cada recurso e proceder à sua classificação na fase de pré-seqüenciamento. Isto permite que o algoritmo de seqüenciamento trabalhe inicialmente apenas com o(s) recurso(s) mais crítico(s), tornando o problema mais simples. Numa etapa posterior, na fase de factibilização, será analisado se a seqüência obtida é factível também do ponto de vista dos demais recursos.

O que se propõe aqui é classificar os recursos em dois tipos, os *recursos críticos*, que são os únicos utilizados na fase de geração das seqüências, e o *recursos não críticos* que só são utilizados na fase de factibilização.

A seguir são analisadas três abordagens diferentes de se fazer esta classificação: a classificação baseada na quantidade total demandada de cada recurso, a classificação baseada na demanda instantânea acumulada devido a coexistência de operações na planta e a classificação baseada nos conceitos de criticalidade/crucialidade.

3.3.2.1 Classificação baseada na demanda total de recursos

A classificação baseada na demanda total de recursos é obtida a partir de uma estimativa, para cada recurso, do tempo mínimo necessário para a execução de todas as tarefas, supondo a utilização máxima do recurso. Esse tempo é calculado a partir da quantidade do recurso necessária para a execução das operações de todas as tarefas e da oferta do recurso.

Para cada recurso tem-se que a demanda total para executar todas as operações de uma determinada tarefa é dado por:

$$D^r = \sum_{i}^{N} \sum_{j}^{M} Q_{ij}^r \times t_{ij} \tag{3.8}$$

onde :

D^r = demanda total do recurso r para a execução de todas as tarefas

Q_{ij}^r = variável agregada que representa o consumo do recurso r pela tarefa i no processador j

t_{ij} = variável agregada que representa o tempo de processamento total da tarefa i no processador j

O tempo mínimo necessário para a execução de todas as tarefas é obtido através da compatibilização entre a demanda total de recursos e a oferta de recursos e pode ser obtido através da seguinte equação :

$$T^r = \frac{D^r}{q^r} \tag{3.9}$$

onde :

T^r = tempo mínimo necessário para a execução de todas as tarefas

q^r = variável agregada que representa a oferta do recurso r

A ideia de definir a criticalidade de um recurso a partir da quantidade total demandada por todas as tarefas, vem do fato que se um determinado recurso impuser um tempo mínimo necessário para realizar as tarefas maior que os demais, este será também o tempo total para os demais recursos, e então a quantidade total dos demais recursos será mais facilmente acomodada neste tempo que é maior do que os respectivos mínimos. A figura 3.4, ilustra isto: o valor do T^r obtido para o recurso 1 (T^1) é maior que o valor obtido para o recurso 2 (T^2), portanto aquela quantidade necessária para o recurso 2 que antes era concentrada no intervalo 0 e T^2, agora poderá ser acomodada dentro de um novo intervalo 0 e T^1, que é o intervalo mínimo necessário para a alocação das tarefas, e como ele é maior, provavelmente o recurso 2 não será decisivo na definição da seqüência final a ser obtida.

Exemplo : Suponha um caso no qual existam 4 tarefas com 3 operações, e que cada uma das operações necessite de 3 recursos para a sua completa execução. Considere a oferta dos recursos constante no tempo e com valor unitário de 10, 12, 12 respectivamente. Considere também que o consumo de uma operação é constante durante a sua execução.

Desta forma, para os seguintes dados relativos aos tempos de processamento e ao consumo das operações para os três recursos :

$$t_{ij} = \begin{vmatrix} 8 & 9 & 5 \\ 6 & 7 & 7 \\ 7 & 7 & 12 \\ 4 & 7 & 12 \end{vmatrix}, Q_{ij}^1 = \begin{vmatrix} 8 & 6 & 4 \\ 6 & 2 & 4 \\ 7 & 3 & 6 \\ 3 & 4 & 8 \end{vmatrix}, Q_{ij}^2 = \begin{vmatrix} 4 & 2 & 4 \\ 5 & 3 & 4 \\ 2 & 6 & 3 \\ 4 & 2 & 2 \end{vmatrix}, Q_{ij}^3 = \begin{vmatrix} 4 & 3 & 4 \\ 2 & 5 & 3 \\ 3 & 3 & 4 \\ 2 & 4 & 3 \end{vmatrix}$$

teremos os seguintes valores para T^r = 49, 24, 25, para r = 1, 2 e 3 respectivamente.

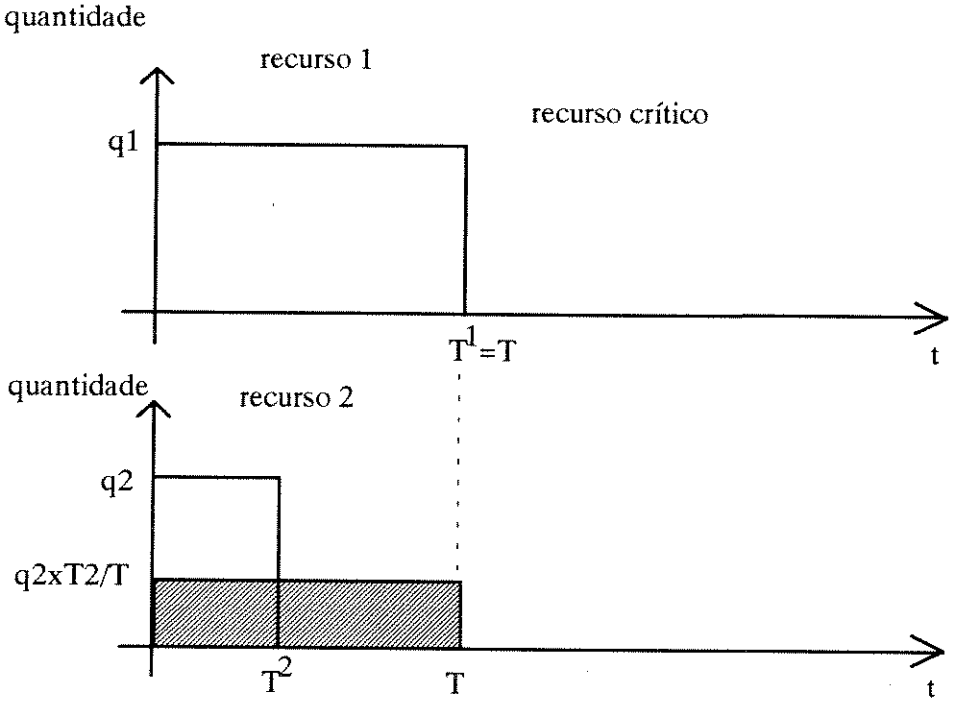


Figura 3.4 : Gráfico comparativo da quantidade necessária de recursos

Uma outra forma se obter uma estimativa do "makespan", é através da utilização de "heurísticas" bastante conhecidas para o flowshop, como as heurísticas SMBB ("Single Machine Based Bound") e FMBB ("Full Machine Based Bound") (Baker (1974) e Rodrigues (1992)). Caso o valor do "makespan" calculado por uma destas heurísticas seja muito maior que o valor do max{ T^r }, obviamente os recursos não serão críticos, pois haverá um tempo folgado para o seu consumo. O algoritmo proposto considera também esta possibilidade, mas não é o caso de interesse no trabalho.

3.3.2.2 Classificação baseada na demanda instantânea originada pela coexistência de operações

A classificação dos recursos como proposto na seção anterior não leva em consideração a possível demanda instantânea acumulada pela coexistência de operações na planta. Esta acumulação não poderá exceder a oferta instantânea de recursos, portanto a oferta limita a coexistência de operações. Uma coexistência de um número menor de operações implica, em geral, em um "makespan" maior, ou seja, os recursos que "limitam mais" as coexistências de operações são os que tem maior influência na definição do "makespan", e portanto são aqueles que devem obrigatoriamente ser considerados no processo de busca.

A coexistência de operações que ocorrerá em qualquer solução do problema de seqüenciamento é variável, i.e., podem ocorrer diferentes quantidades de operações coexistindo ao longo do tempo. A figura 3.5 ilustra esta situação num caso simples: existem instantes de tempo em que apenas uma operação está sendo processada, instantes em que coexistem duas e instantes que coexistem três operações.

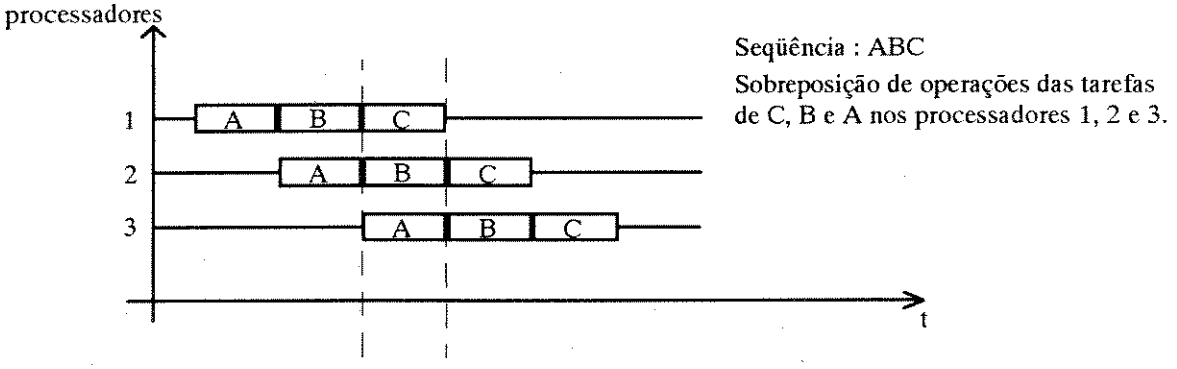


Figura 3.5 : Sobreposição de operações devido a seqüência

Não é possível classificar os recursos em função das coexistências que cada um deles permitirá na solução do problema e, portanto, no seu impacto sobre o "makespan", porque as coexistências que vão existir serão de diferentes tipos e não previsíveis a priori. Porém parece óbvio que algum tipo de análise possa ser feita numa fase de pré-seqüenciamento já que os dados de consumo individual de cada uma das operações estão disponíveis.

O que se propõe é um procedimento heurístico baseado nos conjuntos de possíveis coexistências de M, M-1, M-2, ..., 1 operações de diferentes tarefas para cada um dos recursos. É importante notar que na construção destes conjuntos não se leva em

consideração que as ofertas dos recursos podem inviabilizar algum de seus elementos (coexistências). Justamente esta informação é a que será utilizada para classificar os recursos.

Dois parâmetros sobre os conjuntos citados acima podem ser obtidos facilmente :

- o valor médio de consumo de cada recurso por todas as possíveis coexistências de M, M-1, ..., 1 operações, que tem a expressão analítica mostrada a seguir; e

- a porcentagem de possíveis coexistências de M, M-1, ..., 1 operações que implicam num consumo acumulado maior que a oferta do recurso r. Esta porcentagem é obtida por enumeração e cálculo do consumo de todas as coexistências.

A média de consumo acumulado pode ser contraposta à oferta de um recurso de tal forma a se obter um índice que possa ser comparado aos índices calculados para os demais recursos, definindo então, a partir desta comparação, o grau de criticalidade de cada um deles. Já as porcentagens das coexistências de operações que possuam um consumo acumulado maior que a oferta podem ser comparadas diretamente entre os recursos e a partir desta comparação se definir os recursos mais críticos.

O procedimento heurístico proposto está baseado em considerar um recurso mais crítico que o outro, se a oferta instantânea do primeiro permitir uma coexistência máxima de operações em número menor que o segundo, ou seja, está se admitindo que se o número de operações que podem coexistir é menor, provavelmente o "makespan" resultante será suficientemente grande para viabilizar a alocação das operações satisfazendo o consumo do segundo recurso, dado que este permite uma coexistência maior de operações.

A. Cálculo da média do consumo acumulado pela coexistência de operações

A média de consumo acumulado pela coexistência de operações é calculada a partir da soma dos consumos das operações que podem ser processadas simultaneamente na planta.

O número total de coexistências de M-x operações (sem repetições) pertencentes a N tarefas diferentes é dado por $T_{M\text{-}x}$ (equação 3.10).

$$T_{M-x} = C_{M,M-x} \times A_{N,M-x}, \text{ p/ } N \geq M \tag{3.10}$$

onde:

$C_{N,M} = \dfrac{N!}{M!(N-M)!}$, combinações de N tarefas em M processadores e

$A_{N,M} = \dfrac{N!}{(N-M)!}$, arranjos de N tarefas em M processadores.

A quantidade de vezes que cada consumo deve ser considerado para o cálculo do consumo total do recurso r pelas coexistências de M-x operações pertencentes a N tarefas pode ser escrito como:

$$V_{M-x} = C_{M-1,M-(x+1)} \times A_{N-1,M-(x+1)} \tag{3.11}$$

Desta forma a média de consumo do recurso r pelas coexistências de M-x operações na planta é dado por:

$$U^r_{M-x} = \frac{V_{M-x}}{T_{M-x}} \times \sum_{i=1}^{N} \sum_{j=1}^{M} q^r_{ij}, \text{ para x = 0,1,...,M-1.} \tag{3.12}$$

Estas equações podem ser simplificadas e chegaremos ao seguinte:

i)para x = 0, temos que M-x = M e a equação 3.12 pode ser escrita como

$$U^r_M = \frac{\dfrac{(M-1)!}{(M-1)!\times 1!} \times \dfrac{(N-1)!}{(N-M)!}}{\dfrac{M!}{M!\times 0!} \times \dfrac{N!}{(N-M)!}} \times \sum_{i=1}^{N} \sum_{j=1}^{M} q^r_{ij} \tag{3.13}$$

simplificando teremos:

$$U^r_M = \frac{\sum_{i=1}^{N} \sum_{j=1}^{M} q^r_{ij}}{N} \quad \text{,e} \tag{3.14}$$

ii)fazendo o mesmo para x = 1,2,..,M-1 teremos que

$$U^r_{M-x} = \frac{M-x}{M} \times U^r_M, \text{ para } 0 < x < M \text{ e } N \geq M. \tag{3.15}$$

B. Cálculo das porcentagens das coexistências de operações que possuem um consumo acumulado maior que uma certa porcentagem da oferta

O cálculo destas porcentagens, ao contrário do cálculo da média que pode ser obtido analiticamente, é feito através da enumeração de todas as possíveis coexistências das operações na planta. O cálculo reduz-se à soma dos consumos individuais, este cálculo será realizado para T_{M-x} coexistências (para cada x).

A título ilustrativo 20 tarefas a serem seqüenciadas em 4 processadores terão um total de 116.280 possíveis coexistências de M operações e 50 tarefas em 4 processadores 5.5×10^6 possíveis coexistências, o que corresponde a um tempo de 0.12 e 5.5 segundos respectivamente em um microcomputador PC 386 DX com 40 Mhz, o qual realiza 10^6 operações semelhantes a este procedimento por segundo.

C. Exemplo :

Utilizando-se os dados do exemplo apresentado na seção 3.3.2.1 temos como resultado para esta classificação os seguintes valores :

	Média do consumo			Porcentagem acima da oferta		
Rec.	M	M-1	M-2	M	M-1	M-2
1	15.25	10.17	5.08	87.50	44.44	0
2	10.25	6.83	3.42	20.83	0	0
3	10.00	6.67	3.33	4.17	0	0

Na tabela apresentada acima é possível verificar que o recurso 1 possui um valor percentual de coexistência de M e M-1 operações na planta bastante superior aos demais, o que o caracterizaria como um recurso crítico. A comparação da média das coexistências com o valor da oferta dos recursos 10, 12, e 12 para os recursos 1, 2, e 3 respectivamente também leva à mesma conclusão.

A seguir será apresentada uma nova forma de se classificar os recursos baseada nos conceitos de criticalidade de uma operação apresentados no capítulo anterior. Esta classificação, que apesar de não ter sido utilizada na implementação final do algoritmo, serve de alternativa à classificação apresentada nesta seção, fato este que será discutido mais adiante.

3.3.2.3 Classificação baseada na criticalidade de uma operação

Curvas de criticalidade são normalmente utilizadas para se determinar a criticalidade das operações (Keng (1988) e Sycara (1991)). Estas curvas são baseadas no tempo de processamento das operações e em janelas de tempos definidas pelo par de datas de início mais cedo e fim mais tarde das operações. Para a utilização desse tipo de abordagem para a classificação de recursos propõe-se aqui acrescentar ao cálculo das criticalidades, variáveis que exprimem a quantidade necessária de recursos para a execução das operações, bem como a quantidade ofertada.

O conceito de criticalidades utilizado é baseado na proposta de Keng (1988), onde a criticalidade de uma operação é calculada a partir da seguinte equação:

$$c_{ij} = \frac{t_{ij}}{DW_{ij}} \tag{3.16}$$

onde :

c_{ij} = criticalidade da operação j da tarefa i

DW_{ij} = janela de tempo para a realização da operação j da tarefa i

t_{ij} = tempo da operação j da tarefa i

A definição das janelas de tempo só pode ser feita definindo um tempo final para execução de todas as operações de todas as tarefas. Para fins de classificação na fase de pré-seqüenciamento tem-se uma primeira estimativa deste tempo dada por :

$$T = max\{max[T^r], T_{min}, \underset{i}{max}[\sum_{j} t_{ij}]\} \tag{3.17}$$

onde :

T = horizonte para o cálculo da criticalidade de uma operação

T_{min} = tempo mínimo para execução de todas as tarefas obtida pela "heurística" FMBB

O terceiro termo da equação 3.17 se faz necessário porque as parcelas anteriores não garantem que o horizonte obtido seja factível, podendo levar a valores de DW negativos. Isto pode acontecer porque o T^r não leva em conta o encadeamento das operações e o T_{min} calculado pela heurística FMBB é apenas uma estimativa do tempo mínimo necessário para a execução das operações.

As janelas de tempo das operações são calculadas considerando-se o encadeamento existente entre elas nos diversos processadores. Desta forma, o cálculo dos DW_{ij} para as operações de uma tarefa é dado por :

$$DW_{ij} = T - \sum_{k=1}^{j-1} t_{ik} - \sum_{k=j+1}^{M} t_{ik} = T - \sum_{k=1}^{M} t_{ik} + t_{ij} \qquad (3.18)$$

A figura 3.6 mostra a construção das janelas de tempo para uma tarefa com três operações. Os tempos de processamento total das operações 1, 2 e 3 são dados por 5, 7 e 6 u.t. respectivamente, e o tempo total para construção das janelas é dado por T = 30 u.t.

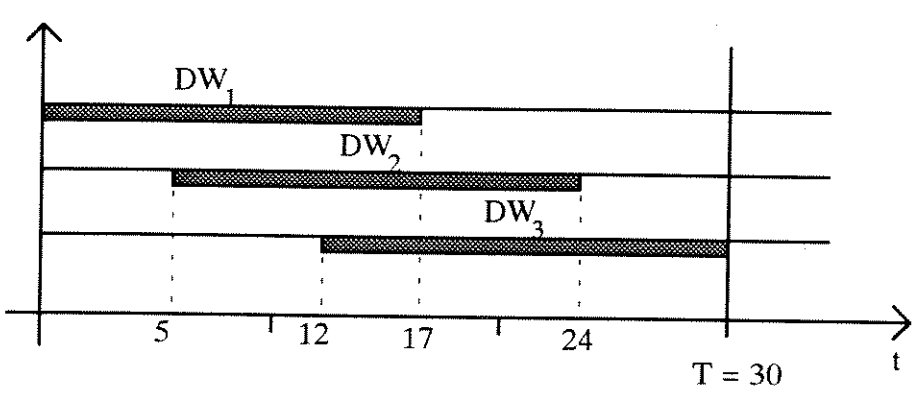


Figura 3.6 : Janela de tempo (DWs) para uma tarefa

Keng (1988) e Sycara (1991) consideram para o cálculo da criticalidade apenas recursos com oferta e demanda unitárias, conseqüentemente para a consideração de valores não unitários é necessário alterar a forma de se obter este valor. O que se propõe é estender a definição de criticalidade, utilizando como base a equação de Keng, acrescentado-se variáveis que representam a oferta e o consumo de recursos. Desta forma, tem-se que:

$$C_{ij}^{r} = \frac{t_{ij}}{DW_{ij}} \times p_{ij}^{r}, \qquad (3.19)$$

onde :

p_{ij}^{r} = fator de ponderação que representa a demanda total de recursos dividida pela oferta dado por:

$$p_{ij}^{r} = \frac{Q_{ij}^{r}}{q^{r}} \qquad (3.20)$$

O valor da criticalidade obtido desta forma, não representa uma demanda firme pelo recurso. Isto acontece porque o DW, para o qual este valor é calculado, é em geral maior que o tempo efetivo necessário para a execução das operações, e também pela ponderação proposta. Esta ponderação faz com que o valor da criticalidade seja diminuído proporcionalmente pela oferta do recurso, e portanto os recursos que tiverem uma oferta maior terão este valor diminuído proporcionalmente.

A comparação entre os diversos recursos, para efeito da classificação, é feito através de curvas de crucialidades. Estas curvas mostram o nível de competitividade por um determinado recurso, e são obtidas a partir da soma das criticalidades obtidas para os intervalos de tempo associados às operações de uma mesma tarefa. Os valores de cada instante de tempo destas curvas são obtidas através da expressão $\sum_i \sum_j \left\{ I\,(Q_{ij}^r / q^r) \times t_{ij} \,] / DW_{ij} \right\}$, onde o DW_{ij} (janela de tempo da operação j da tarefa i) cobrir o instante de tempo.

Quanto maior for o valor da crucialidade, mais crítico será considerado o recurso. A figura 3.7 mostra a construção de uma curva de crucialidade a partir de três DWs: DW1, DW2, DW3 os quais correspondem as operações 1, 2 e 3 de uma dada tarefa. Os valores das criticalidades para as três operações são os seguintes: 0.5, 0.3, 0.4 respectivamente.

Este tipo de abordagem tem o inconveniente de que os valores das curvas de crucialidade não são obtidos de forma analítica, sendo necessário a construção das curvas de criticalidade e da posterior soma dos valores instante a instante, para obtenção das mesmas, ou seja, é necessário construir a solução do problema. A vantagem em relação à anterior é que a dimensão do problema não cresce exponencialmente com a quantidade de tarefas e nem de processadores, colocando-se como alternativa para a anterior quando estes valores se tornarem críticos.

Exemplo :

Utilizando-se os dados do exemplo apresentado no item A temos como resultado para esta classificação os seguintes valores :

Recurso	Crucialidade
1	1.4314
2	0.7285
3	0.7518

Uma discussão mais detalhada sobre a utilização destas diferentes propostas para classificação de recursos será apresentada no capítulo referente aos exemplos de aplicação.

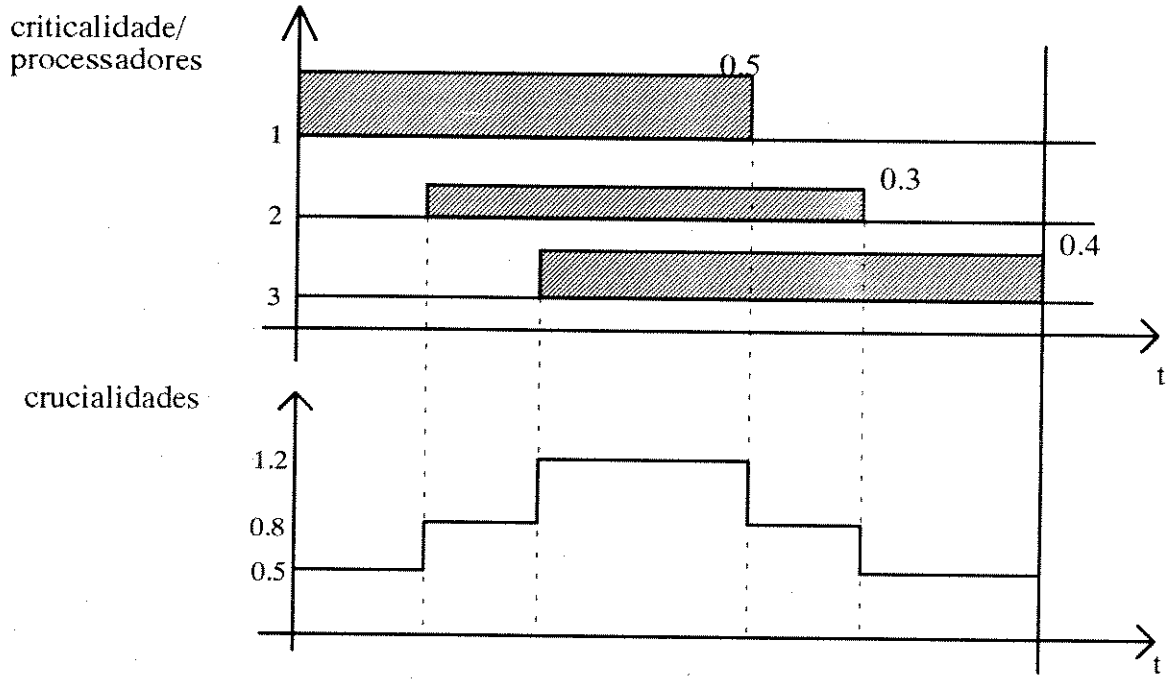


Figura 3.7 : Curvas de crucialidades

3.3.3 Seqüenciamento: Processo de busca orientado pelas restrições

O processo de busca na fase de geração de seqüências é realizado utilizando um algoritmo de BAB chamado de C*, o qual incorpora as principais características dos algoritmos propostos originalmente por Rodrigues (1992) e estendidos por Campos (1993) para considerar a alocação externa de tarefas. É utilizado para gerar as seqüências um modelo agregado da planta, onde as operações de preparação, processamento e transferência são considerados como um único bloco, considerando apenas os recursos críticos.

Rodrigues propôs uma série de algoritmos chamados de A, B, C1 e C2, já apresentados na seção 2.4.2. Os algoritmos A e B são considerados algoritmos básicos, a diferença entre eles está no cálculo da parcela referente à estimava do valor do "lower-

bound" - lb. O algoritmo A considera no cálculo do lb que o perfil ofertado na região II está disponível integralmente, já o algoritmo B através de um mecanismo de "look-ahead", verifica o instante de início mais cedo que as operações iniciais das próximas tarefas a serem seqüenciadas podem efetivamente começar, o instante t^*_{K+1}. É a partir deste instante que é feita a estimativa do lb possibilitando assim uma melhor estimativa deste valor. Resultados de simulação mostram que o tempo total gasto pelos dois algoritmos é bastante semelhante, pois se por um lado o algoritmo A faz menos cálculos, por outro, a estimativa do lb é inferior ao do algoritmo B, conseqüentemente um maior número de nós é explorado na busca. Os algoritmos C1 e C2 correspondem ao algoritmo A e B com o acréscimo na estimativa do lb no cálculo do valor associado ao encadeamento das operações a serem seqüenciadas, utilizando a "heurística" FMBB (Rodrigues (1992) e Rodrigues e outros(1993)).

O algoritmo C2, como discutido acima, faz uma melhor estimativa do valor do lb em cada nó quando comparado ao C1, trazendo como consequência uma exploração mais eficiente da árvore de busca através da redução da quantidade de nós a serem pesquisados. O inconveniente deste algoritmo reside no fato de que para fazer uma melhor estimativa do valor do lb de um nó, é realizada uma explosão deste nó para pesquisar o instante inicial mínimo no qual a primeira operação de seus filhos pode começar, e isto implica no dispêndio de um tempo maior. Em alguns nós o instante inicial encontrado pelo algoritmo C2 coincide com o do algoritmo C1, ou seja, t^*_{K+1}= C(K,1), o que nos leva a concluir que a união destes dois algoritmos pode ser uma estratégia interessante. Neste caso, o algoritmo resultante atuaria como C1 quando possível e como C2 nos demais casos.

A seguir é apresentada uma proposta que engloba a utilização dos algoritmos C1 e C2, propostos por Rodrigues, em um só algoritmo, o algoritmo C*. Este algoritmo atua na expansão de um nó como o algoritmo C1 quando for encontrado um t^*_{K+1} que seja igual a C(K,1), e como o algoritmo C2 quando nenhum t^*_{K+1} for igual a C(K,1).

Algoritmo C*

Como o algoritmo C1 é mais rápido do ponto de vista dos cálculos a serem realizados do que o algoritmo C2, é interessante que valores de início da primeira operação muito próximos a C(K,1), i.e., t^*_{K+1} é muito próximo de C(K,1), sejam considerados para efeito de opção entre os algoritmos como sendo iguais a C(K,1), e portanto o algoritmo C1 seja escolhido. Neste caso para determinar quando valores próximos a C(K,1) devam ser considerados, será utilizando um pequeno Δt a ser adicionado a C(K,1), comparado-se então este valor com t^*_{K+1}. Isto deve ser feito, pois o

ganho na estimativa do valor do lb não justificaria a quantidade de cálculos a serem feitos pelo algoritmo C2. A seguir será mostrado o algoritmo C* e a utilização do Δt.

A estratégia proposta é a seguinte :

1) Inspecionar no conjunto U de tarefas não seqüenciadas, qual tarefa apresenta o menor consumo de recursos na primeira operação. Supondo que seja $\alpha \in U$

$Q_{\alpha 1} = \min_{i \in U} \{Q_{i1}\}$;

2) Em seguida calcular : t_{α}^{*};

3) Se $t_{\alpha}^{*} \leq C(K,1) + \Delta t$ então calcular LB segundo o algoritmo C1;

4) Se $t_{\alpha}^{*} > C(K,1) + \Delta t$ então iniciar o cálculo e a verificação de $t_{m,K+1}^{*}$ para as demais tarefas não seqüenciadas. A primeira tarefa que apresentar $t_{m,K+1}^{*} \leq C(K,1) + \Delta t$ parar e calcular o LB segundo o algoritmo C1;

5) Se para todo o conjunto U : $t_{m,K+1}^{*} > C(K,1) + \Delta t$ então usar o algoritmo C2;

6) Fim.

O processo de busca termina com a obtenção da seqüência ótima, mas as demais seqüências completas e parciais com os respectivos valores de "makespan" e "lower-bound" são mantidos em uma lista, com a finalidade de possibilitar uma retomada do processo de busca, caso seja necessário.

3.3.4 Seqüenciamento: Análise de factibilidade

A simplificação utilizada na solução do problema na fase de geração de seqüências impõe que se submeta a solução encontrada pelo BAB a uma análise para a verificação da factibilidade da seqüência quando se considera o modelo detalhado e os demais recursos.

Na fase de factibilização é utilizado um modelo detalhado, e o objetivo é verificar a factibilidade da seqüência obtida na fase anterior em relação a este modelo considerando todos os recursos. Para que esta seqüência após a factibilização possa ser considerada ótima, é necessário comparar o valor final do "makespan" resultante da factibilização, caso o mesmo tenha sido alterado, com os demais valores de "makespan" ou "lower-bound"-lb das seqüências completas ou parciais, respectivamente, que ainda não tenham sido submetidas à fase de factibilização. Caso existam seqüências com algum desses valores inferiores ao da seqüência analisada, deve haver um retrocesso à fase de geração de seqüências. Se no retrocesso houver alguma seqüência completa com valor do "makespan"

inferior ao "makespan" da seqüência factibilizada, esta é então escolhida e passa-se à fase de factibilização, caso contrário retoma-se o processo de busca.

O que se faz então é manter o estado atual da busca na fase de geração de seqüências, através de uma lista, a qual contém todas as seqüências parciais e completas com os respectivos valores do lb no caso de seqüências parciais e do "makespan" no caso de seqüências completas.

A análise de factibilidade consiste de uma simulação do processamento das tarefas nos diversos processadores, através do lançamento das tarefas seqüencialmente na planta considerando as operações de preparação, processamento e transferência independentemente. O lançamento das tarefas implica na programação horária (construção da carta de Gantt) e dos respectivos gráficos de capacidade associados aos recursos, para cada uma das operações das tarefas.

Os tempos de início determinados anteriormente para esta seqüência que está sendo analisada são desprezados, ou seja, nesta fase apenas a seqüência das tarefas obtidas na fase anterior é utilizada, pois a alocação das operações que antes era feita considerando-se a operação como um único bloco formado pela agregação dos tempos e da quantidade de recurso demandada pelas operações de preparação, processamento e transferência, agora será feito para cada uma delas individualmente.

Como resultado desta fase têm-se para a seqüência um novo valor para o "makespan", agora considerado factível. Este novo valor pode eventualmente ser diferente do "makespan" obtido na fase anterior. Caso este valor seja menor ou igual ao "makespan" anterior esta seqüência é considerada a seqüência final, e o procedimento de busca é encerrado. Caso contrário a busca que havia sido interrompida na fase anterior é retomada.

O procedimento de factibilização é o seguinte :

1. i = 1 (variável que representa as tarefas)
2. j = 1 (variável que representa os processadores)
3. Para a operação de preparação da tarefa i no processador j FAZER :
   A. Determinar o tempo de início mais cedo para a operação no processador j que satisfaça as restrições de precedência temporal e da limitação da oferta dos recursos
   B. Atualizar a carta de Gantt e atualizar os gráficos de capacidade dos recursos

4. Idem para a operação de processamento

5. Idem para a operação de transferência considerando simultaneamente os processadores j e j+1 se este estiver livre ou o processador j e o tanque de armazenagem.

6. Fazer j = j + 1

7. Se j ≤ M retornar a 3. (M quantidade total de processadores)

8. Fazer i = i + 1

9. Se i ≤ N retornar a 2. (N quantidade total de tarefas)

10. Comparar o valor do "makespan" da seqüência detalhada com o "makespan" anterior

SE "makespan" detalhado > "makespan" anterior

ENTÃO reintroduzir a seqüência na lista de seqüências da fase anterior e retomar o procedimento de busca

SENÃO encerrar a fase de factibilização e passar para a fase de pós-seqüenciamento

11. Fim

Uma pequena alteração pode ser feita neste algoritmo para evitar que retrocessos desnecessários sejam realizados. Neste caso o teste da etapa 10 deve ser feito entre o valor do "makespan" obtido para a seqüência na fase de factibilização e o "makespan" da próxima seqüência candidata a ser analisada na fase de seqüenciamento.

3.3.5 Pós-seqüenciamento

Este nível é responsável pela realização de uma análise detalhada da utilização de todo os recursos. Isto é feito a partir do resultado obtido no nível anterior, construindo-se gráficos de capacidade que são utilizados pelo planejador para análise do nível de utilização de cada um deles. Esta análise possibilita, entre outras coisas, a verificação da disponibilidade de recursos que possa eventualmente ser utilizada para a programação de novos produtos não previstos inicialmente, a negociação deste excedente com outros setores da empresa ou a verificação da ociosidade de algum equipamento e planejar uma manutenção preventiva.

Além dos gráficos de capacidade é também fornecido um gráfico de Gantt com todo o detalhamento necessário para que sirva de instrumento para a execução das tarefas planejadas.

Um outro resultado que pode ser obtido a nível de pós-seqüenciamento é a obtenção de seqüências alternativas, geradas durante a fase de seqüenciamento, que eventualmente possam ser utilizadas caso a primeira delas não possa ser cumprida.

Portanto, na fase de pós-seqüenciamento são realizadas as seguintes funções :

1. Detalhamento da carta de Gantt da seqüência final;
2. Construção dos gráficos detalhados de capacidade para todos os recursos; e
3. Definição das seqüências alternativas geradas pelo seqüenciamento.

3.4 Algoritmo multi nível : Descrição global

O objetivo desta seção é dar uma visão geral do algoritmo, discutindo as funções existentes nos diferentes níveis, a interface entre os níveis e a interface com o meio externo (planejador). A figura 3.8 mostra um diagrama de blocos da estrutura multi nível contendo as diversas funções pertencentes a cada um dos níveis. Na figura 3.8 a ligação proveniente do nível de seqüenciamento em direção ao nível de pré-seqüenciamento, está relacionada com uma possível intervenção do planejador na solução do problema, através da alteração de alguns dos dados iniciais, como por exemplo: prazos de entrega, alteração na quantidade de recursos, etc.

A seguir é feita uma descrição sucinta das funções existentes em cada um dos diferentes níveis.

3.4.1 Nível de pré-seqüenciamento

A primeira tarefa deste nível é a entrada de dados referentes ao seqüenciamento que se deseje realizar. Na entrada de dados são fornecidos os seguintes dados :

- quantidade de processadores;
- quantidade e identificação dos recursos;
- perfil de oferta dos recursos;
- quantidade e identificação das tarefas;
- perfil de consumo das operações de preparação, processamento e transferência para cada uma das tarefas;
- tempo de duração das operações; e
- valor do "Beam-search".

Obs.: Caso não se deseje utilizar a "Beam-search", basta utilizar o próprio valor de N - quantidade total de tarefas - como o valor "beam".

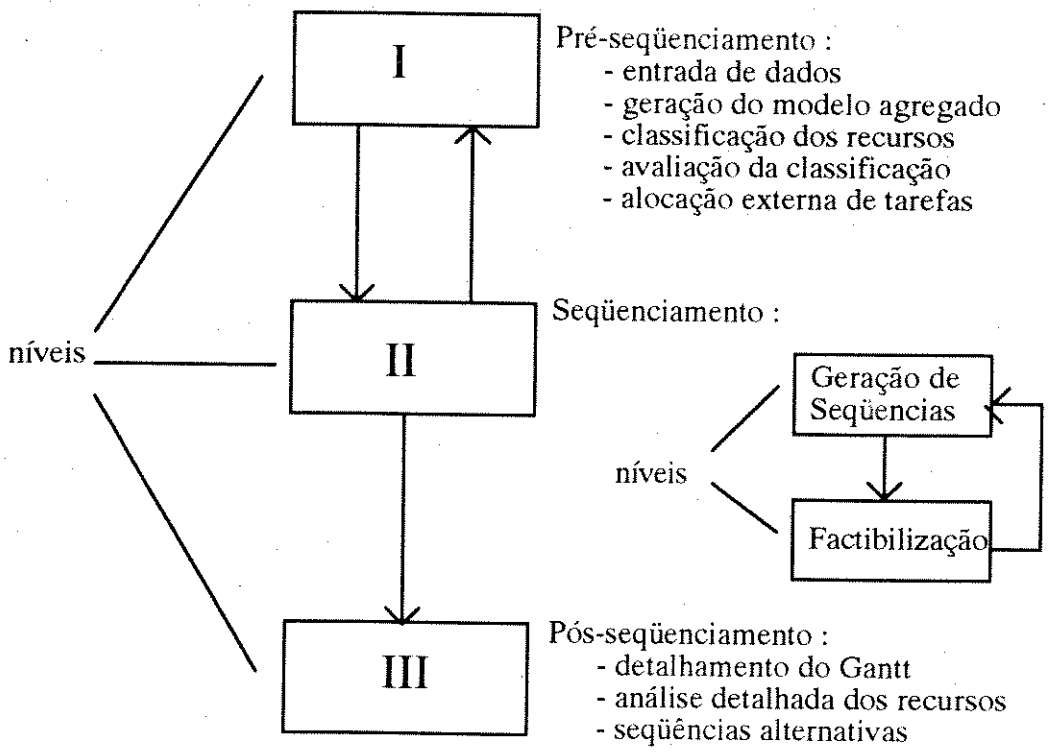


Figura 3.8 : Diagrama de blocos da estrutura multi nível

A classificação dos recursos é realizada através da utilização da abordagem baseada na demanda de recursos, e da abordagem baseada na coexistências de operações. A utilização conjunta destas duas abordagens possibilita ao planejador tomar decisões sobre a classificação dos recursos de uma forma mais eficaz, pois ele pode a partir de uma possível divergência entre as duas abordagens, analisar os resultados numéricos apresentados e tomar sua decisão baseado nesta análise.

A abordagem baseada nos conceitos de criticalidade/crucialidade embora seja uma alternativa interessante, pois nos casos estudados mostrou um comportamento semelhante quanto à classificação dos recursos, tem como desvantagem o fato de que os seus resultados são obtidos de uma forma construtiva, i.e., é necessário que se construam as curvas para a obtenção dos valores das crucialidades máximas. Entretanto, apesar desta abordagem implicar em um processamento mais "pesado" em termos computacionais, ela não tem a desvantagem de uma possível explosão combinatorial da abordagem baseada na coexistência de operações. Portanto, em aplicações mais críticas, pode-se perfeitamente utilizar a classificação baseada na criticalidade/crucialidade como alternativa à abordagem baseada na coexistência de operações.

A avaliação da classificação dos recursos tem por finalidade incluir na solução do problema a possibilidade do planejador, após análise dos resultados da classificação pelos métodos propostos, interferir na classificação caso julgue necessário.

Este tipo de procedimento é importante, pois existem detalhes nos sistemas reais que são difíceis de serem adicionados aos algoritmos, quer seja pela dificuldade de modelagem, quer seja pela dificuldade de se obter este conhecimento dos planejadores.

Existem recursos que pela sua pouca importância, quer seja porque existem em grande quantidade ou pelo seu baixo custo, podem ser considerados com capacidade infinita e, portanto, ficar fora do seqüenciamento.

Desta forma, o resultado da classificação é submetido à apreciação do planejador, que após uma análise destes resultados pode alterar a classificação de um determinado recurso. As seguintes situações são desde já vislumbradas :

- incluir um recurso classificado como não crítico no conjunto dos críticos;
- incluir um recurso classificado como crítico no conjunto dos não críticos; e
- eliminar recursos do seqüenciamento.

Embora este tipo de decisão seja algo de difícil execução, pois uma interferência equivocada pode ter influência negativa na solução do problema, a própria experiência que o planejador pode adquirir com a utilização deste mecanismo, pode fazer com que ele adquira uma melhor sensibilidade sobre a influência dos recursos na fase seguinte. Pode-se desde já vislumbrar a possibilidade da existência de um sistema especialista que auxiliaria e/ou substituiria o planejador nesta tarefa.

A alocação externa de tarefas é um procedimento realizado exclusivamente pelo planejador. A partir das tarefas definidas inicialmente para serem seqüenciadas são selecionadas as tarefas para as quais serão definidas janelas. Para estas tarefas além da identificação são também fornecidas as datas de início mais cedo e fim mais tarde, para que seja possível definir as janelas de duração. Após a definição das datas é verificada se as mesmas são factíveis a partir da verificação deo tamanho da janela da tarefa:

$$DW_i \geq \sum_j t_{ij}$$

onde:

$DW_i = DD_i - RT_i$;

DD_i = prazo de entrega ("Due Date") da tarefa i e

RT_i = tempo de pronto ("Ready Time") da tarefa i.

Caso alguma inconsistência seja verificada, o planejador deve alterar as respectivas datas.

3.4.2 Nível de seqüenciamento

A fase de geração de seqüências é realizada utilizando o algoritmo C*. Nesta fase é utilizado o modelo agregado, visando a obtenção do resultado de uma forma mais eficiente do ponto de vista do tempo de processamento. O resultado desta fase é uma lista ordenada de seqüências a qual representa o estado atual da busca. O ordenamento desta lista é feito através do valor do "lower-bound" da seqüência. A primeira seqüência completa desta lista é a seqüência que vai ser analisada na fase de factibilização e que caso se mostre factível, será também a seqüência final.

A fase de factibilização é realizada através de uma simulação do processo de fabricação das tarefas na planta. Nesta simulação as tarefas que fazem parte da seqüência escolhida na fase de geração de seqüências são "lançadas na planta" uma a uma, respeitando-se as restrições temporais das operações e as restrições a nível de capacidade dos recursos. Caso se detecte uma infactibilidade, o detalhamento da seqüência ultrapassa o valor do "makespan" obtido anteriormente, um novo valor para o "makespan" é calculado, a seqüência é reintroduzida na lista de seqüências respeitando-se o ordenamento da mesma e o procedimento de busca da fase anterior é retomado.

No retorno à fase de geração de seqüências podem ocorrer duas situações distintas:

1) A lista de seqüências possui posicionada em primeiro lugar uma seqüência que está completa, portanto basta que a mesma seja retirada da lista e passa-se a fase de factibilização ou

Após a fase de factibilização e quando houver um retorno à fase anterior, a seqüência que foi submetida ao processo de factibilização é comparada com a seqüência obtida anteriormente (se houver) com a finalidade de se obter um limitante superior ("upper bound") que limitará a expansão de ramos na árvore de busca. Esta seqüência que é factível e ao mesmo tempo a melhor seqüência obtida até o momento é chamada de solução incumbente ("incumbent solution").

2) A lista de seqüências possui posicionada em primeiro lugar uma seqüência ainda parcial, i.e., não foi totalmente explodida, e portanto o algoritmo do BAB é retomado até que se encontre uma nova seqüência completa.

3.4.3 Nível de pós-seqüenciamento

Este nível tem como função fornecer ao planejador uma listagem contendo o detalhamento da programação horária das tarefas, os gráficos de capacidade detalhados para todos os recursos, a programação de utilização da armazenagem intermediária e uma relação contendo as seqüências alternativas geradas durante a solução do problema.

Isto é feito a partir do resultado obtido no nível de seqüenciamento. Esta análise possibilita entre outras coisas, a verificação da disponibilidade de recursos, que possam eventualmente ser utilizados para a programação de novos produtos não previstos inicialmente, ou a negociação deste excedente com outros setores da empresa, ou ainda para verificar a ociosidade de algum equipamento e planejar uma manutenção preventiva.

3.5 Observações finais

Algoritmos de busca em árvore do tipo BAB, apesar de possibilitarem uma enumeração inteligente do espaço de busca, i.e., dirigem a busca pelo valor de uma função de custo associada aos ramos da árvore, pesquisando somente os que tiverem um valor menor que os demais, nem sempre são aplicáveis a problemas de grande dimensão. Desta forma, é importante utilizar mecanismos que possibilitem restringir o espaço de busca. Dentre estes mecanismos podemos citar a utilização de *limitantes superiores* ("upper bounds"), os quais eliminam da árvore de busca ramos com valores da função de custo maiores que estes limitantes. Outro mecanismo é a utilização de *heurísticas* que podem contribuir para uma redução significativa do espaço de busca, como por exemplo uma busca do tipo "Beam-search", na qual apenas os n melhores filhos de um nó permanecem ativos após a sua explosão, neste caso n é chamado de valor beam. O inconveniente de se adicionar heurísticas à busca é que as soluções obtidas podem ser consideradas sub-ótimas, mas para problemas de grande dimensão esta é uma alternativa bastante aceitável.

A agregação de dados tal como foi proposta, na qual os consumos das operações são agregados utilizando-se o valor da média de consumo preservando-se a sua integral, pode fazer com que na fase de factibilização haja uma redução do valor do "makespan". Isto faz com que não possa se garantir a otimalidade do algoritmo. Uma alternativa que se coloca é utilizar os valores mínimos de consumo das operações no lugar da média, desta

forma, os valores do "makespan" das seqüências não poderiam ser reduzidos e conseqüentemente a otimalidade do algoritmo em relação à factibilização estaria garantida. Esta alternativa tem como inconveniente o fato de que haverá um número maior de retrocessos entre a fase de factibilização e a fase de geração de seqüências.

A consideração de tarefas alocadas externamente e o procedimento heurístico para a eliminação de tarefas candidatas consideradas menos críticas propostos por Campos (1993) e apresentados na seção 2.4.3 contribuem também para a redução do espaço de busca. No caso das tarefas alocadas externamente esta redução se deve ao fato de que alguns ramos da árvore só se viabilizam quando o "relógio" do algoritmo coincidir com as janelas de tempo associadas às tarefas. Já o procedimento heurístico elimina diretamente os ramos que seriam originados pelas tarefas consideradas menos críticas. Nos casos examinados por Campos foi verificado que a utilização desta heurística contribui significativamente para a redução do espaço de busca, sem perda da otimalidade do algoritmo.

3.6 Conclusão

Propôs-se um algoritmo para o seqüenciamento de flowshops, o qual considera as subdivisões existentes nas operações de processamento e a limitação na oferta de recursos de uso compartilhados. Foi utilizado como critério para o seqüenciamento a minimização do tempo total para a realização de todas as tarefas (minimização do "makespan").

Alguns autores na literatura questionam o fato de se utilizar o "makespan" como critério de desempenho para algoritmos de seqüenciamento, mas na Indústria de Processos Químicos as instalações industriais são extremamente caras e é usual avaliar o custo de não se produzir a quantidade nominal instalada. Portanto, o critério "makespan" passa a ter um significado especial, pois leva a maximizar a produção e conseqüentemente aumentar a utilização da capacidade instalada.

O algoritmo proposto utiliza uma abordagem multi nível a qual possibilita a consideração de um nível maior de detalhes do que as abordagens ditas globais. Isto é possível por causa da decomposição do problema e da utilização de diferentes níveis de abstração. Com isto é possível tratar de problemas com características mais próximas dos problemas reais existentes e com técnicas que buscam a solução ótima do problema.

Foram propostas uma forma para a agregação de dados e três diferentes formas para se classificar recursos. No entanto o algoritmo pode ser complementado com novas

formas para estes dois tipos de procedimentos, possibilitando desta forma aplicar o algoritmo a situações não previstas originalmente.

Capítulo 4 - Exemplo de aplicação e apresentação de resultados

4.1 Introdução

Neste capítulo são apresentados alguns exemplos de simulação do algoritmo para ilustrar a sua aplicabilidade ao seqüenciamento de flowshops. A seção 4.2 apresenta uma comparação entre as soluções obtidas quando se utiliza a abordagem hierárquica versus a abordagem global, tomando como base o mesmo algoritmo e variando-se a quantidade de recursos considerados críticos. Esta seção tem como objetivos demonstrar a viabilidade da abordagem multi nível e validar as propostas de classificação apresentadas no capítulo anterior. Nesta seção é utilizado apenas o modelo agregado.

A seção 4.3 apresenta os resultados de aplicação da metodologia como um todo, considerando os modelos agregados e detalhados, assumindo que a classificação dos recursos sugerida pelo algoritmo é a correta, fato que é verificado na seção 4.2. Nesta seção são também apresentados resultados de simulação quando se utiliza a "Beam-search".

O algoritmo utilizado para fazer as simulações é o algoritmo C* apresentado no capítulo anterior. Não são considerados nos exemplos apresentados alocações externas de tarefas.

4.2 Comparação entre as soluções obtidas por uma abordagem global e pela abordagem hierárquica no tratamento de recursos compartilhados

Este seção tem os seguintes objetivos: i) fazer uma análise comparativa entre o desempenho da abordagem global onde são considerados simultaneamente todos os recursos, e a abordagem hierárquica onde os recursos críticos são considerados na fase de geração de seqüências e os demais apenas na fase de factibilização; e ii) discutir as diferentes formas de se classificar os recursos, definindo os recursos críticos que serão considerados na fase de geração de seqüências.

Para a realização destas análises foi utilizada uma versão do programa, a qual utiliza o modelo agregado tanto na fase de geração de seqüências como na fase de factibilização. Isto é necessário para que se possa através da avaliação dos resultados referentes a frequência de retrocessos ("backtrackings"), fazer uma análise exclusivamente em função dos recursos considerados nas duas fases, sem influência do maior nível de

detalhes existentes no modelo detalhado que se utilizará normalmente na fase de factibilização. Este outro aspecto será considerado na seção 4.3.

Serão utilizados para a classificação dos recursos dois critérios, o critério 1, no qual a classificação é baseada na demanda total de recursos apresentado na seção 3.3.2.1, e o critério 2, no qual a classificação é baseada na demanda instantânea originada pela coexistência de operações na planta apresentado na seção 3.3.2.2.

A proposta para a classificação dos recursos em críticos ou não críticos para os critérios 1 e 2 é a seguinte:

- Critério 1 - Serão considerados críticos os recursos cujo valor do T^r for maior que a média de todos os T^r, ou seja:

Se $T^r > \frac{1}{R}\sum_{r=1}^{R} T^r$ *então* o recurso é classificado como crítico

senão o recurso é classificado como não crítico.

- Critério 2 - A classificação dos recursos será baseada numa análise do percentual cumulativo das coexistências de operações na planta. Serão feitas análises sobre os percentuais de coexistências de M, M-1, ..., 2 operações cujo consumo acumulado ultrapassa o valor da oferta de recursos. A classificação dos recursos em críticos e não críticos será discutida nos exemplos.

Nos exemplos de simulação, além dos resultados do seqüenciamento, serão também apresentadas informações sobre o tempo gasto pelo algoritmo para encontrar a solução utilizando a abordagem global onde todos os recursos são considerados como críticos e o tempo gasto para obtenção da seqüência pelo algoritmo proposto segundo a abordagem hierárquica para diferentes classificações. A partir destes tempos será feita uma comparação entre o desempenho destas duas abordagens.

Nos casos apresentados a seguir será apresentado o desempenho do algoritmo e da classificação dos recursos utilizando-se diferentes combinações de classificação. É importante salientar que na prática não se simulam diversas situações com diferentes classificações, mas sim apenas uma, e portanto é fundamental que a classificação proposta pelo algoritmo seja decisiva na identificação dos recursos críticos.

4.2.1 Exemplos de simulação

São apresentados nesta seção três casos de testes os quais utilizam o mesmo conjunto de dados representando um flowshop com três processadores, com cinco recursos de uso compartilhados (utilidades), com seis tarefas a serem seqüenciadas, todas disponíveis no instante inicial, e sem limite na data final. A oferta e o consumo de recursos são considerados constantes, o que simplifica bastante a análise dos exemplos. Os valores das quantidades dos recursos ofertadas nos três casos serão diferentes, e apresentados em cada um deles. Os dados dos exemplos de simulação são apresentados na tabela 4.1.

As tabelas 4.2 e 4.3 apresentam respectivamente os valores das médias de consumo acumuladas pela coexistência de operações na planta e a quantidade total de coexistências. Este valores são válidos para os três casos a serem apresentados, pois independem da oferta dos recursos.

Tabela 4.1 : Dados para os exemplos de simulação

Tarefa A	Processadores		
	1	2	3
Tempos	8	9	5
Recursos	Consumos		
1	8	6	4
2	4	2	4
3	4	3	4
4	8	5	5
5	5	6	3

Tarefa B	Processadores		
	1	2	3
Tempos	6	7	7
Recursos	Consumos		
1	6	2	3
2	5	3	4
3	4	5	2
4	5	3	4
5	2	5	3

Tarefa C	Processadores		
	1	2	3
Tempos	7	7	12
Recursos	Consumos		
1	7	3	6
2	3	4	5
3	6	3	3
4	3	5	3
5	4	6	5

Tarefa D	Processadores		
	1	2	3
Tempos	7	6	5
Recursos	Consumos		
1	4	5	4
2	2	6	3
3	3	4	4
4	2	6	3
5	5	3	4

Tarefa E	Processadores		
	1	2	3
Tempos	4	7	12
Recursos	Consumos		
1	3	4	8
2	5	4	5
3	4	3	6
4	5	6	5
5	6	4	3

Tarefa F	Processadores		
	1	2	3
Tempos	6	8	7
Recursos	Consumos		
1	3	4	3
2	5	3	2
3	4	5	3
4	6	5	4
5	2	4	3

O objetivo destes três exemplos de simulação é o de ilustrar o ganho em tempo de processamento que pode ser obtido através de uma escolha cuidadosa dos recursos críticos a serem considerados na fase de geração de seqüências. Dentro do intervalo de criticalidade dos recursos a classificação dos recursos como críticos e não críticos daqueles situados nos extremos é clara. Os exemplos mostram porém que a discriminação numa faixa intermediária é difícil e uma definição errada pode levar a uma grande perda da eficiência do algoritmo. Isto leva à proposta feita nas conclusões desta seção de utilizar uma estratégia baseada na eliminação apenas dos recursos claramente não críticos.

Tabela 4.2 - Valores das médias de consumo acumulado pela coexistência de M - x operações na planta.

Recurso	Valor da média		
	M	M-1	M-2
1	13.83	9.22	4.61
2	11.50	7.67	3.83
3	11.67	7.78	3.89
4	13.83	9.22	4.61
5	12.17	8.11	4.06

Tabela 4.3 - Quantidade total de M-x coexistências.

M	M-1	M-2	Total
120	90	18	228

A. Caso 1: Dois recursos críticos. Análise da escolha total e da escolha isolada de cada um deles.

Este exemplo tem o objetivo de discutir as diferentes formas de se classificar os recursos e de mostrar o desempenho do algoritmo quando se escolhem dois dos recursos críticos ou cada um deles individualmente e compará-los à abordagem global, onde todos os recursos são considerados conjuntamente. A tabela 4.4 mostra os resultados obtidos para a classificação dos recursos.

Tabela 4.4 - Resultados obtidos para a classificação do caso 1

Rec	Oferta	Critério 1		Critério 2						
		média = 42.8		Total de coexistências			% de coexistências			
		T^r	Clas.	M	M-1	M-2	M	M-1	M-2	Clas.
1	13	48	crit.	66	5	0	55.0	5.5	0	crit.
2	12	41	não	38	0	0	31.7	0	0	não
3	13	39	não	19	0	0	15.8	0	0	não
4	13	45	crit	67	2	0	55.8	2.2	0	crit.
5	13	41	não	35	0	0	29.2	0	0	não

Na tabela 4.4 as colunas indicam:

Rec - identificação do recurso

Oferta - oferta do recurso r por unidade de tempo.

Resultados obtidos pelo critério 1

média - valor da média dos valores dos T^r utilizada para classificar os recurso em críticos ou não críticos. Se T^r > média o recurso é considerado crítico.

Valor do T^r - Tempo mínimo necessário para executar todas as tarefas considerando a quantidade ofertada do recurso r.

Clas. - classificação dos recursos sob o critério 1.

Resultados obtidos pelo critério 2

total de coexistências - total de coexistências de M-x operações cujo consumo acumulado é maior que a oferta do recurso r.

% de coexistências - porcentagem de coexistências de M-x operações cujo consumo acumulado é maior que a oferta do recurso r. O total de coexistências é dado na tabela 4.3.

Clas. - classificação dos recursos sob o critério 2.

A tabela 4.5 apresenta os resultados obtidos em função do conjunto de recursos definidos como críticos. A comparação é feita com a situação em que todos os recursos são considerados críticos.

Tabela 4.5- Resultados de simulação para o caso 1

Classificação dos recursos 1 2 3 4 5	Quantidade de retrocessos	Nós sondados	Tempo gasto no BAB [seg]	Tempo gasto na Fact. [seg]	Comparação valor %
0 0 0 0 0	0	142	2.97	0.00	100 %
0 1 1 0 1	6	174	2.04	0.11	72.4 %
0 1 1 1 1	30	248	1.97	0.60	86.5 %
1 1 1 0 1	233	602	4.37	4.35	293.6 %

Na tabela 4.5 as colunas indicam:

Classificação dos recursos - cada uma das colunas representa a classificação do recurso : 0 - recurso crítico e 1 - recurso não crítico.

Quantidade de retrocessos - quantidade de vezes que o algoritmo retornou a fase de geração de seqüências após a fase de factibilização.

Nós sondados - quantidade de nós sondados durante a fase de busca.

Tempo gasto no BAB - tempo total gasto durante o algoritmo de busca.

Tempo gasto na factibilização - tempo total gasto na fase de factibilização.

Comparação valor % - comparação do tempo total gasto para a obtenção da solução para cada uma das possíveis classificações em relação a abordagem onde todos os recursos são considerados críticos, utilizando a seguinte expressão : (Valor/Valor de referência)x100, onde Valor de referência é o valor obtido com todos os recursos definidos como críticos, classificação (0 0 0 0 0).

Resultados :

Seqüência final : E C F B A D

Makespan = 70 [u.t.]

Comentários:

Nos resultados de simulação apresentados nas tabelas 4.4 e 4.5 é possível verificar que a solução encontrada utilizando a classificação dos recursos 1 e 4 como críticos e os demais como não críticos, classificação (0 1 1 0 1), apresentou um desempenho melhor do que a solução obtida pela abordagem global onde todos os recursos são considerados como críticos, classificação (0 0 0 0 0). A primeira gastou apenas 72,4% do tempo gasto pela segunda, mesmo tendo sido realizados 6 retrocessos para a obtenção da solução;

As curvas apresentadas na figura 4.1 auxiliam na interpretação e definição da criticalidade relativa dos recursos. Estas curvas representam o percentual cumulativo da

ocorrência de coexistências de M-x operações na planta. No eixo das abcissas estão representados os valores de consumo acumulado pela coexistência de M-x operações, e no eixo das ordenadas uma escala de probabilidade normal. Na figura as letras (a), (b), (c), (d), e (e) representam os recursos 1, 2, 3, 4, e 5 respectivamente. Os gráficos envolvidos por um quadro mais acentuado identificam os recursos críticos.

Analisando estas curvas é possível verificar que o recurso 1 é um recurso crítico, pois a sua oferta, representada na figura pela linha vertical que passa pelo valor 13 no eixo horizontal, mostra que as coexistências de M operações na planta são drasticamente restringidas por este recurso. Se passarmos agora a analisar apenas as demais coexistências de M-1 e M-2, veremos que mesmo assim este ainda é o recurso que mais restringe a ocorrência de coexistências de M-1 operações. O mesmo comentário pode ser feito em relação ao recurso 4.

Dentro de uma ótica de escolha dos recursos críticos, e não da eliminação dos recursos não críticos, caberia agora perguntar-se se devem ser escolhidos como críticos os recursos 1, 4, ou 1 e 4. Na Tabela 4.5 são apresentados os resultados para estas classificações, o primeiro considerando os recursos 1 e 4 como críticos, classificação (0 1 1 0 1), o segundo considerando apenas o recurso 1 como crítico, classificação (0 1 1 1 1) e o terceiro considerando apenas o recurso 4 como crítico, classificação (1 1 1 0 1).

O desempenho da solução encontrada utilizando a classificação (0 1 1 1 1) é bom (86.5%) mesmo que ligeiramente inferior ao obtido com a classificação (0 1 1 0 1) que considera os recursos 1 e 4 como críticos (72.4%). A outra opção (1 1 1 0 1), considerando apenas o recurso 4 como crítico é totalmente insatisfatória (293.6%). Observando os índices de criticalidade dados na tabela 4.4 (T^r e % de coexistências) nota-se que não existe uma diferença grande entre os recursos 1 e 4 que permita explicar esta diferença de desempenho. Tudo indica então que uma estratégia baseada na eliminação de recursos não críticos (2, 3 e 5 claramente) é mais robusta.

Ainda em relação aos recursos não críticos 2, 3 e 5 é possível verificar na figura 4.1 que a coexistência de M-1 operações não sofrem qualquer tipo de restrição pela quantidade ofertada de recursos. Isto é um indicador bastante forte da não criticalidade dos mesmos.

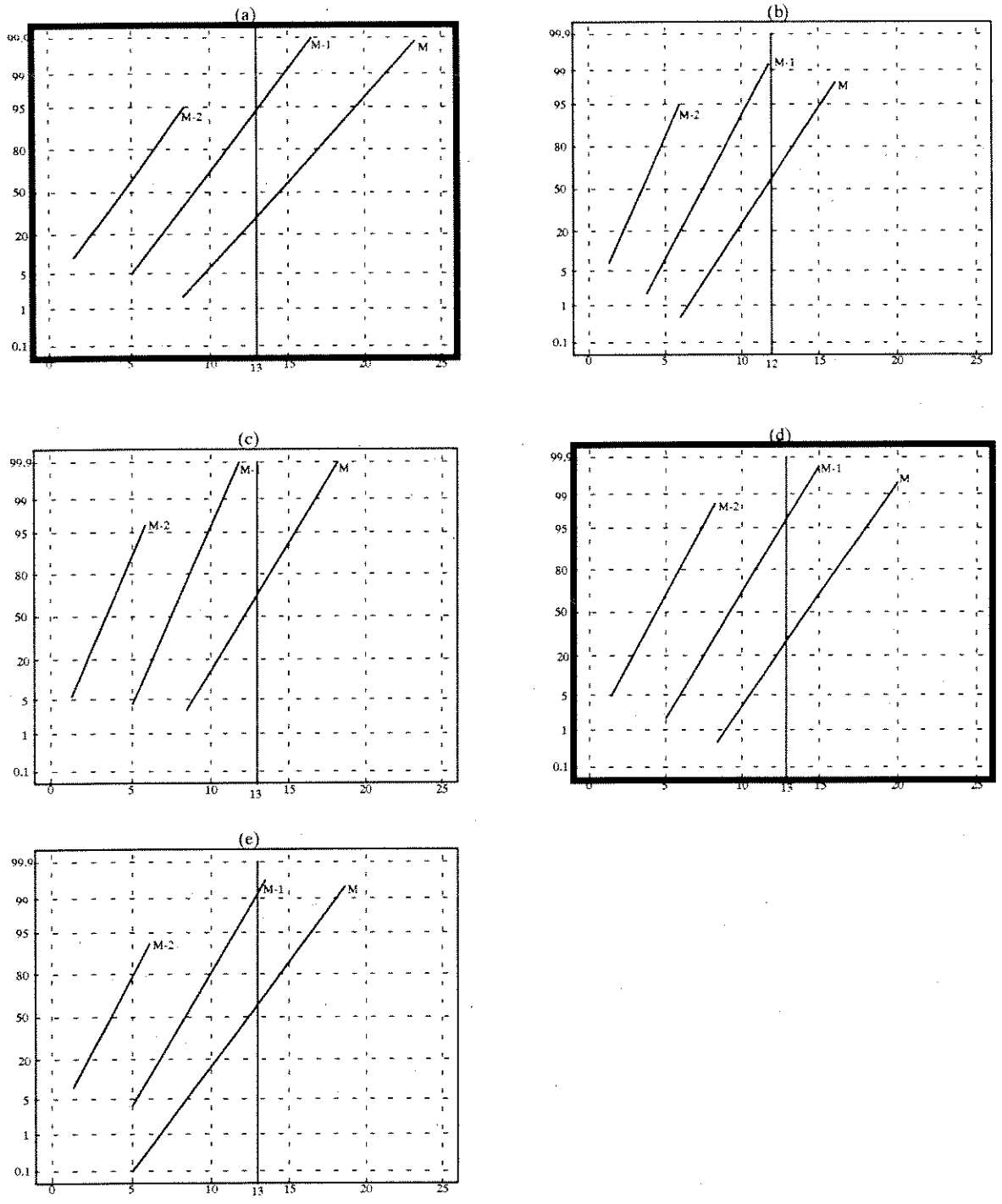


Figura 4.1 : Percentual cumulativo dos recursos caso 1

Já para a classificação que considera apenas o recurso 4 como crítico, classificação (1 1 1 0 1), os resultados apresentados na tabela 4.5 mostram que o desempenho do algoritmo fica bastante comprometido, gastando cerca de três vezes o tempo gasto pela abordagem global. Isto acontece por causa da quantidade de nós sondados, 602 no total, e da quantidade de retrocessos, 233.

É importante notar que o resultado do seqüenciamento não muda em função da escolha da classificação, o que muda é o desempenho do algoritmo.

B. Caso 2 : Três recursos críticos. Análise da escolha total e subconjuntos

Este exemplo, onde três dos cinco recursos são considerados críticos, tem o objetivo de mostrar o desempenho do algoritmo quando se escolhem três recursos críticos ou subconjuntos deles e compará-los entre si e com a abordagem global. A tabela 4.6 mostra os resultados obtidos para a classificação dos recursos. Neste caso foram alterados apenas os valores da oferta dos recursos mantendo-se os demais dados do caso anterior.

Tabela 4.6 - Resultados obtidos para a classificação do caso 2

Rec	Oferta	Critério 1		Critério 2						
		média = 48.8		Total de coexistências			% de coexistências			
		T^r	Clas.	M	M-1	M-2	M	M-1	M-2	Clas.
1	14	45	não	49	2	0	40.8	2.2	0	não
2	12	41	não	38	0	0	31.7	0	0	não
3	10	51	crit.	86	5	0	71.7	5.6	0	crit.
4	11	54	crit.	100	9	0	83.3	10.0	0	crit.
5	10	53	crit.	97	8	0	80.8	8.9	0	crit.

Resultados :

Seqüência final : E C D A F B

Makespan = 73 [u.t.]

Comentários:

Neste caso como pode ser visto na tabela 4.7, a solução encontrada pelo algoritmo quando o resultado da classificação é seguido, classificação (1 1 0 0 0), o desempenho do algoritmo é bem superior quando comparado à abordagem global, classificação (0 0 0 0 0), e que neste caso a quantidade de nós sondados foi a mesma (114 nós), o que nos leva concluir que o algoritmo percorreu o mesmo caminho na árvore de busca, ou seja, pesquisou o mesmo espaço de solução. Neste caso é possível verificar também que não houveram retrocessos, o que significa em outras palavras, que a seqüência final obtida na fase de geração quando submetida a verificação de factibilidade no nível seguinte foi considerada factível. Portanto, os recursos considerados não críticos não tem nenhuma influência no resultado do seqüenciamento.

Tabela 4.7- Resultados de simulação para o caso 2

Classificação dos recursos 1 2 3 4 5	Quantidade de retrocessos	Nós sondados	Tempo gasto no BAB [seg]	Tempo gasto na Fact. [seg]	Comparação valor %
0 0 0 0 0	0	114	2.58	0.00	100 %
0 1 0 0 0	0	114	2.26	0.00	87.6 %
1 1 0 0 0	0	114	1.92	0.00	74.4 %
1 1 0 0 1	1	172	2.20	0.00	85.3 %
1 1 0 1 0	13	207	2.41	0.35	107.0 %
1 1 0 1 1	133	517	3.65	2.76	248.4 %
1 1 1 0 0	0	176	2.20	0.00	85.3 %
1 1 1 0 1	61	348	2.58	1.46	156.6 %
1 1 1 1 0	101	427	3.03	2.33	207.8 %

A escolha de dois recursos como críticos do total de três leva a um desempenho bom em duas situações (1 1 0 0 1) e (1 1 1 0 0), ainda que pior do que a situação anterior onde todos os recursos críticos são escolhidos para a fase de geração de seqüências (1 1 0 0 0). Já a escolha de um único recurso como crítico leva a situações totalmente insatisfatórias.

Analisando-se as curvas apresentadas na figura 4.2 verifica-se que os recurso 3, 4 e 5 são os que mais restringem a ocorrência de coexistência de M operações e portanto são considerados críticos. Para os demais recursos, 1 e 2, observa-se que o recurso 2 é claramente um recurso não crítico pois a ocorrência de M-1 coexistências de sobreposições de operações não sofre qualquer tipo de restrição, já que os recursos críticos praticamente inviabilizam a coexistência de M operações. O outro recurso, recurso 1, embora também classificado como não crítico está numa faixa de transição entre os recursos claramente críticos (3, 4, 5) e o recurso não crítico (2). A tabela 4.7 mostra para o caso (0 1 0 0 0) que o desempenho é bom (87.6%), o que reforça a estratégia proposta de eliminação conservadora de recursos não críticos.

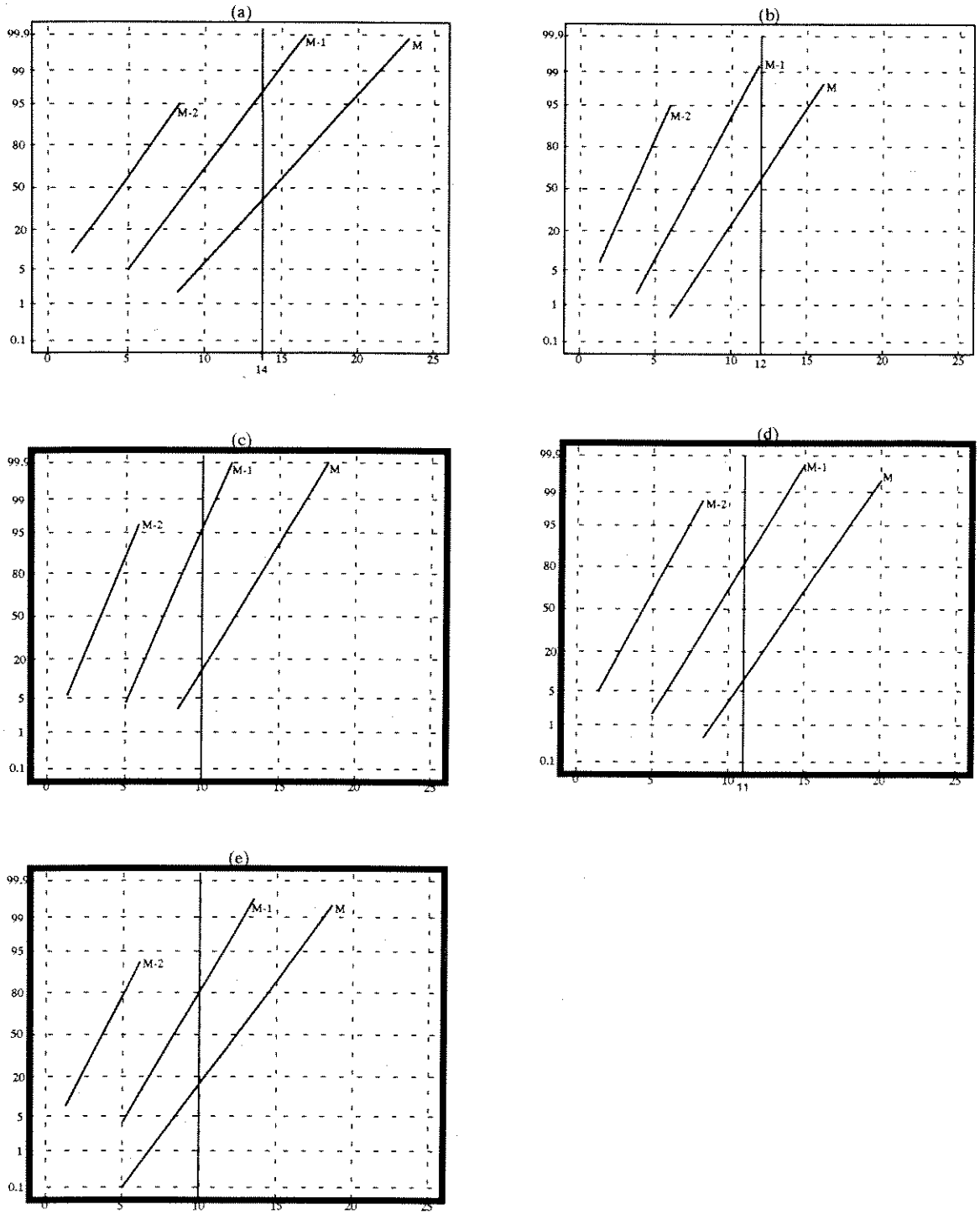


Figura 4.2: Percentual cumulativo dos recursos caso2

C. Caso 3 : Dois recursos críticos pelo critério T^r e três recursos críticos pelo critério de coexistências.

Neste caso, a exemplo do caso anterior, foram alterados apenas os valores da oferta dos recursos, com o objetivo de fazer uma análise de um caso onde os critérios de classificação não coincidem. A tabela 4.8 mostra os resultados obtidos.

Tabela 4.8 - Resultados obtidos para a classificação do caso 3

		Critério 1		Critério 2						
		média = 51.4		Total de coexistências			% de coexistências			
Rec	Oferta	T^r	Clas.	M	M-1	M-2	M	M-1	M-2	Clas.
1	10	63	crit.	102	28	0	85.0	31.1	0	crit.
2	10	49	não	79	5	0	65.8	5.6	0	?
3	12	42	não	39	0	0	32.5	0	0	não
4	10	59	crit.	112	22	0	93.3	24.4	0	crit.
5	12	44	não	51	0	0	42.5	0	0	não

Resultados :

Seqüência final : A D B C F E

Makespan = 83 [u.t.]

Tabela 4.9- Resultados de simulação para o caso 3

Classificação dos recursos 1 2 3 4 5	Quantidade de retrocessos	Nós sondados	Tempo gasto no BAB [seg]	Tempo gasto na Fact. [seg]	Comparação valor %
0 0 0 0 0	0	161	3.40	0.00	100 %
0 0 1 0 1	0	166	2.58	0.00	75.9 %
0 1 1 0 1	0	166	2.14	0.00	62.9 %
0 0 1 1 1	9	235	2.90	0.12	88.8 %
1 0 1 0 1	81	437	4.78	1.45	183.2 %
0 1 1 1 1	14	258	2.53	0.11	77.7 %
1 0 1 1 1	719	1235	6.78	14.86	636.5 %
1 1 1 0 1	88	451	3.84	1.54	158.2 %

Comentários:

Neste caso, como mostram os resultados das tabelas 4.8 e 4.9, a classificação do recurso 2 pelo critério 2 daria alguma margem de dúvida se este deveria ou não ser considerado crítico, pois o recurso 2 apresenta um percentual alto de coexistência de M operações cerca de 65.8%. Os resultados de simulação realmente mostram que este recurso não deveria ser considerado crítico, pois quando a classificação na qual apenas os recursos 1 e 4 são considerados críticos, classificação (0 1 1 0 1), o desempenho do algoritmo é superior ao desempenho de quando se considera também o recurso 2, classificação (0 0 1 0 1), 62,9% no primeiro caso contra 75,8% do segundo em relação à abordagem global.

A tabela 4.9 mostra que a escolha dos recursos 1 e 4 como críticos leva ao melhor desempenho. O acréscimo do recurso 2 ao conjunto dos recursos críticos leva a uma pequena queda no desempenho (devido ao maior número de cálculos no BAB e não ao número de nós sondados). A escolha de um único recurso como crítico pode levar a um bom desempenho (0 1 1 1 1) devido ao menor tempo gasto nos cálculos do BAB, apesar dos retrocessos, ou a um desempenho muito ruim (1 1 1 0 1). A estratégia proposta de uma opção conservadora para a classificação dos recursos críticos a partir da eliminação dos não críticos se mostra realmente como mais robusta, pois se considerar um recurso não crítico como crítico leva sempre a um desempenho do algoritmo melhor do que a solução global, enquanto que o contrário, i.e., considerar um recurso crítico como não crítico pode levar a resultados em termos de desempenho bastante ruins.

A figura 4.3 mostra graficamente os resultados apresentados na tabela 4.8. Nesta figura observa-se que a oferta dos recurso 1 e 4 praticamente inviabilizam a ocorrência de M coexistências de operações. Desta forma, a escolha do recurso crítico se daria a partir da análise da coexistência de M-1 operações.

Conclusões

A metodologia proposta mostra-se eficiente na diminuição do tempo de processamento com redução da ordem de 25% nos exemplos analisados. A escolha de recursos a serem considerados na fase de geração de seqüências (BAB) deve ser cuidadosa porque a não inclusão de um recurso crítico pode dar lugar a um desempenho ruim. Propõe-se então uma estratégia baseada na eliminação dos recursos claramente não críticos, pois desta forma é possível garantir que o desempenho do algoritmo será sempre

melhor que o desempenho da abordagem global onde todos os recursos são considerados críticos.

Nas próximas seções será considerado que a classificação proposta pelo algoritmo é a classificação adequada e somente os resultados para esta classificação serão apresentados.

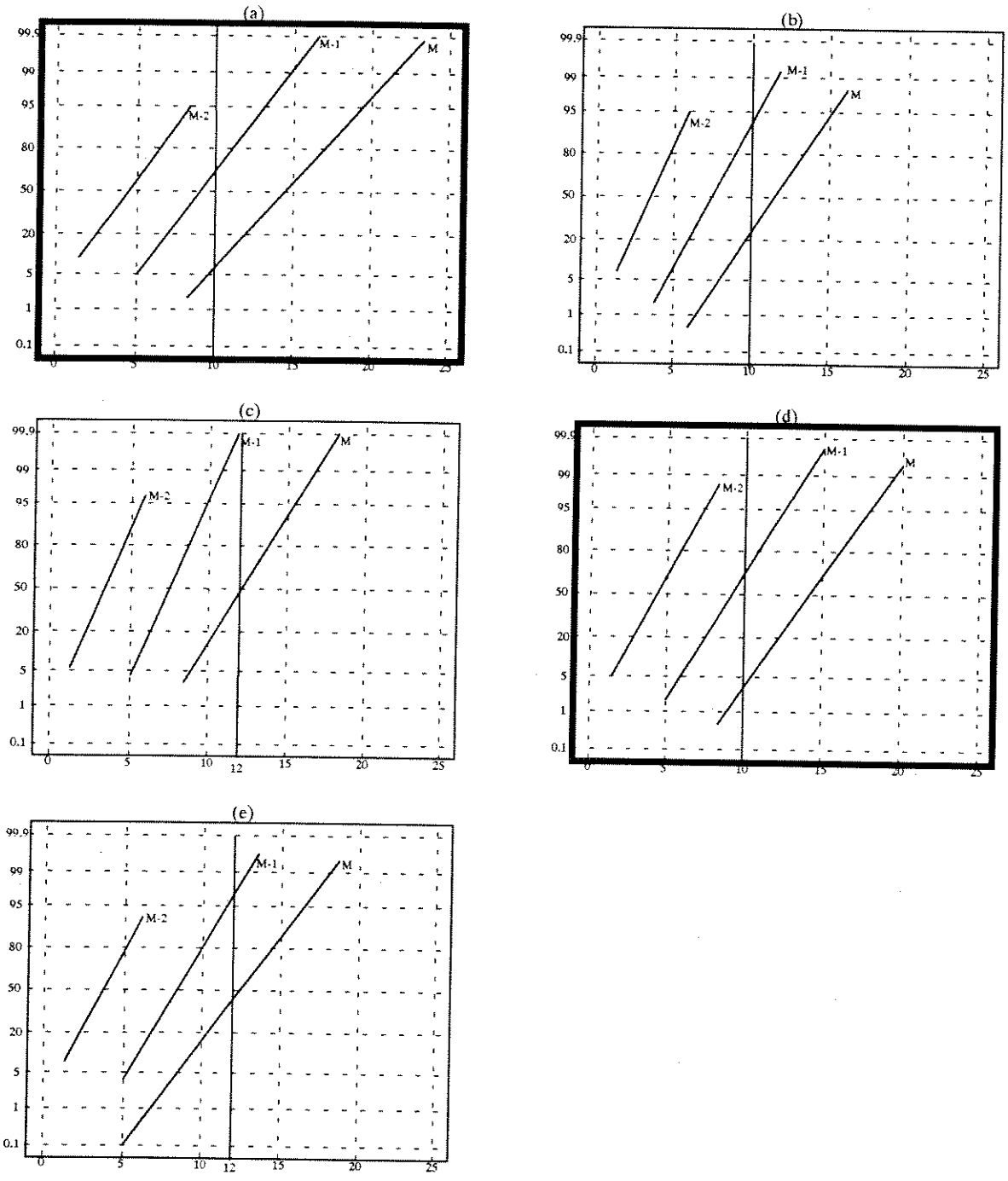


Figura 4.3 : Percentual cumulativo dos recursos caso 3

4.3 Apresentação de resultados utilizando o modelo detalhado

Nesta seção são apresentados exemplos de aplicação da metodologia utilizando um modelo agregado na fase de geração de seqüências e um modelo detalhado na fase de factibilização. Serão apresentados inicialmente três situações diferentes, que contemplam o seguinte: i)consumos idênticos nas operações de preparação, processamento e transferência, ii)consumos das operações de preparação e transferência menores que os respectivos consumos das operações de processamento e iii)consumos das operações de preparação e transferência maiores que os respectivos consumos das operações de processamento.

O objetivo deste conjunto de exemplos é mostrar o desempenho do algoritmo em relação a agregação de dados quando submetidos as três situações citadas acima. A figura 4.4 mostra de forma esquemática estas situações.

É considerado para a realização destes exemplos, um flowshop com três processadores, um conjunto de quatro tarefas e três recursos de uso compartilhado. O cálculo do valor do T^r e do percentual de coexistência de operações com consumo acumulado maior que a oferta é realizado sobre os valores agregados.

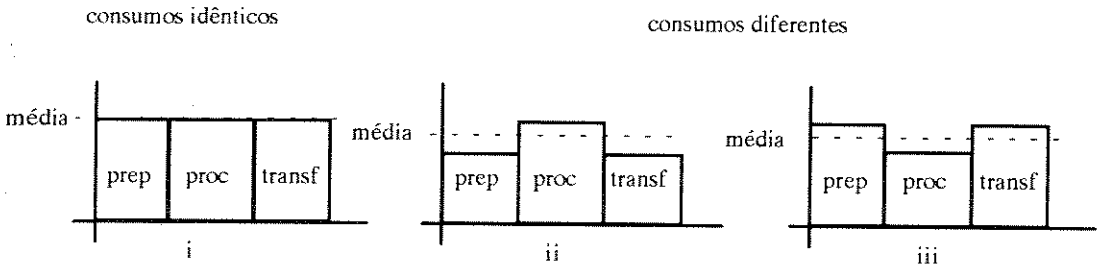


Figura 4.4 : Representação esquemática do consumo das operações - [u.r]x[u.t.]

4.3.1 Exemplo com consumos de recursos idênticos para as três operações

Os dados do exemplo são os seguintes :

Tabela 4.10 - Dados do exemplo

Tarefa A	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	5	6	2	1	1	1
Recursos									
1	8	6	4	8	6	4	8	6	4
2	4	2	4	4	2	4	4	2	4
3	4	3	4	4	3	4	4	3	4

Tarefa B	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	3	4	4	1	1	1
Recursos									
1	6	2	4	6	2	4	6	2	4
2	5	3	4	5	3	4	5	3	4
3	2	5	3	2	5	3	2	5	3

Tarefa C	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	4	4	9	1	1	1
Recursos									
1	7	3	6	7	3	6	7	3	6
2	2	6	3	2	6	3	2	6	3
3	3	3	4	3	3	4	3	3	4

Tarefa D	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	1	4	9	1	1	1
Recursos									
1	3	4	8	3	4	8	3	4	8
2	4	2	2	4	2	2	4	2	2
3	2	4	3	2	4	3	2	4	3

As quantidades de recurso ofertada para os recursos 1, 2 e 3 são respectivamente 10, 12 e 12 unidades de recursos. Os valores agregados são os seguintes :

Tabela 4.11 : Valores agregados

	Tempos			Consumo recurso 1			Consumo recurso 2			Consumo recurso 3		
	1	2	3	1	2	3	1	2	3	1	2	3
A	8	9	5	8	6	4	4	2	4	4	3	4
B	6	7	7	6	2	4	5	3	4	2	5	3
C	7	7	12	7	3	6	2	6	3	3	5	4
D	4	7	12	3	4	8	4	2	2	2	4	3

Os valores obtidos para a classificação de recursos podem ser vistos nas tabelas 4.12 e 4.13 :

Tabela 4.12 -Valores da média de consumo

Recurso	Média		
	M	M-1	M-2
1	15.25	10.17	5.08
2	10.25	6.83	3.62
3	10.25	6.83	3.62

Tabela 4.13- Resultados da classificação dos recursos

Recurso	Oferta	T^r	% de coexistências			Classificação
			M	M-1	M-2	
1	10	49.40	87.50	44.44	0	crítico
2	12	24.58	20.83	0	0	não crítico
3	12	25.75	4.17	0	0	não crítico

Os resultados finais obtidos para este exemplo considerando como crítico o recurso 1 são os seguintes:

Seqüência : A B C D
Makespan = 64 [u.t.]
Nós sondados = 11
Retrocessos = 1.

A tabela 4.14 apresenta os tempos de início e fim das operações para a construção da carta de Gantt apresentada na figura 4.6.

Note pela carta de Gantt da figura 4.6 que embora tenha sido verificada a necessidade da utilização da armazenagem intermediária pela tarefa D, como esta é a última tarefa programada, a mesma pode permanecer no próprio processador até o instante inicial de sua programação no próximo processador. A figura 4.5 mostra o estado final da busca após a obtenção da seqüência final, os nós escurecidos não foram visitados.

Tabela 4.14 : Tabela de tempos para a construção da carta de Gantt

Tarefa	Processador	Tempos de início			
		Transferência	Processamento	Transferência	Preparação
A	1	0	1	6	7
	2	6	8	14	15
	3	14	15	17	18
B	1	16	17	20	21
	2	20	21	25	26
	3	25	26	30	31
C	1	22	32	36	37
	2	36	37	41	42
	3	41	42	51	52
D	1	38	39	40	41
	2	43	44	48	49
	3	53	54	63	64

Utilização de armazenagem intermediária
- produto D - início : 40 - fim : 44
- produto D - início : 48 - fim : 54

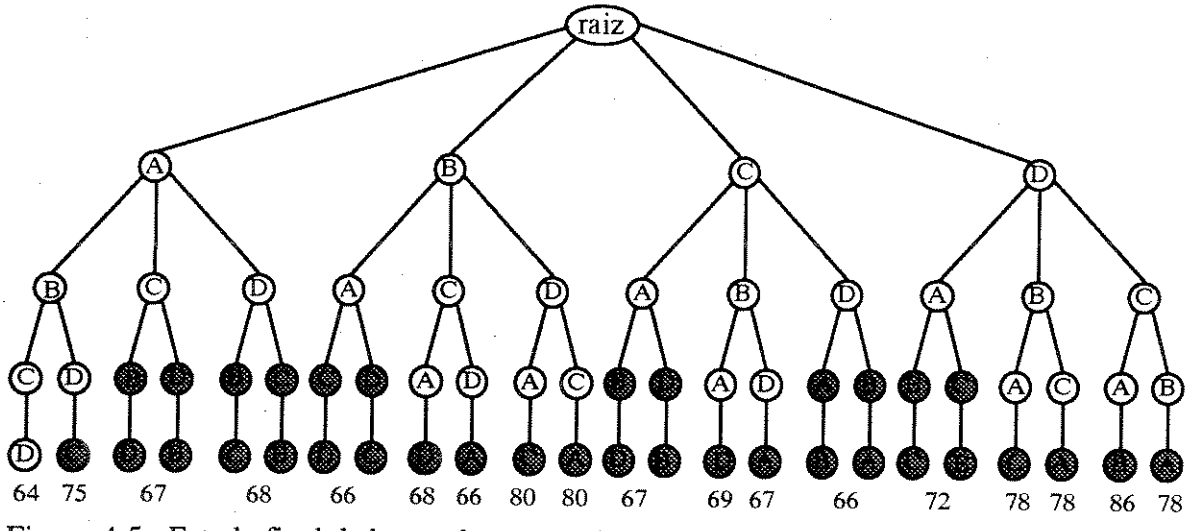
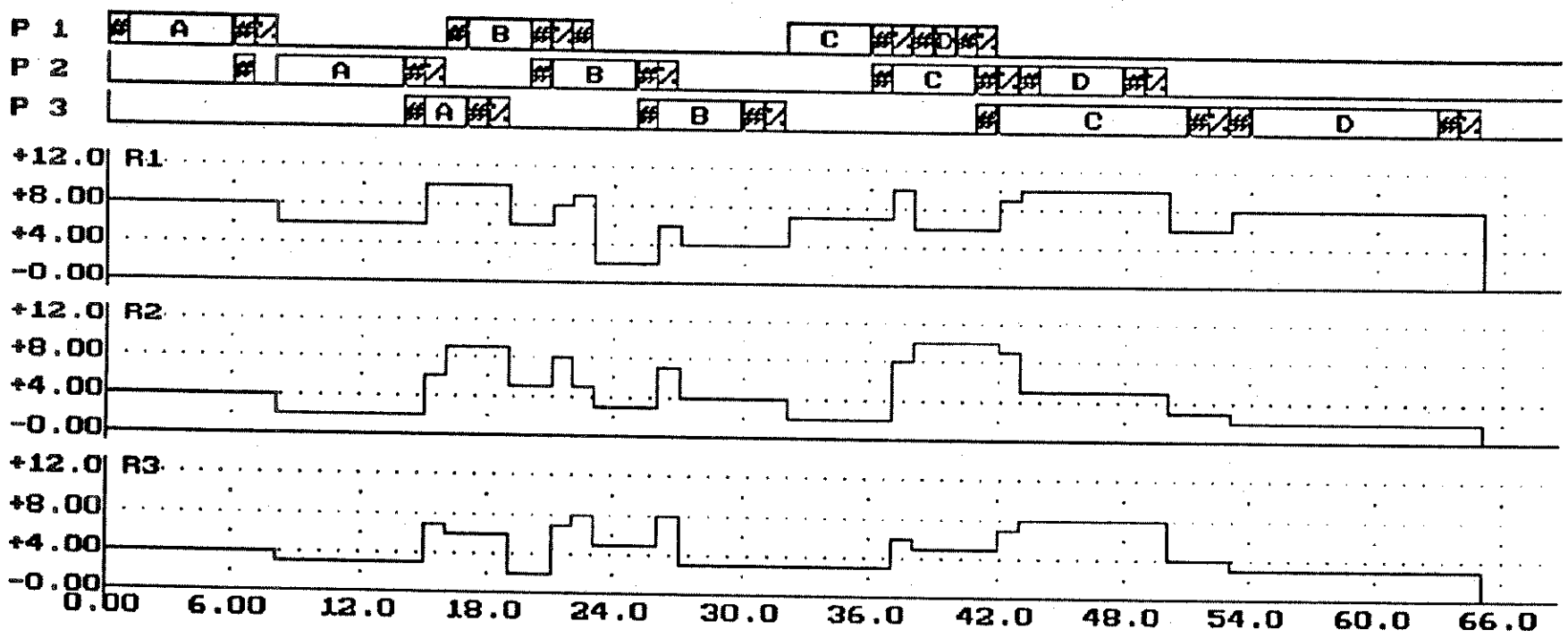


Figura 4.5 : Estado final da busca do seqüenciamento de A B C D.



LEGENDA:

# Transferência	Px - Processador x
% Preparação/Limpeza	Ry - Recurso y
X Processamento da tarefa x	eixo x - [u.t.]
# X # % Tarefa x	eixo y - [u.r.]

Figura 4.6 - Carta de Gantt da seqüência ABCD

4.3.2 Exemplo com consumos das operações de preparação e transferência menores que os respectivos consumos das operações de processamento

Neste exemplo são mantidos os tempos das operações e os valores do consumo da operação de processamento do exemplo anterior. Os consumos das operações de preparação e transferência são reduzidos para valores próximos a metade. O objetivo é o de verificar se a agregação de dados é adequada para esta situação.

Os dados do exemplo são apresentados na tabela 4.15.

Tabela 4.15 - Dados do exemplo

Tarefa A	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	5	6	2	1	1	1
Recursos									
1	2	3	2	8	6	4	2	3	2
2	2	1	2	4	2	4	2	1	2
3	2	1	2	4	3	4	2	1	2

Tarefa B	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	3	4	4	1	1	1
Recursos									
1	3	1	2	6	2	4	3	1	2
2	3	1	2	5	3	4	3	1	2
3	1	3	1	2	5	3	1	3	1

Tarefa C	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	4	4	9	1	1	1
Recursos									
1	3	2	3	7	3	6	3	2	3
2	1	3	2	2	6	3	1	3	2
3	1	2	2	3	3	4	1	2	2

Tarefa D	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	1	4	9	1	1	1
Recursos									
1	1	2	4	3	4	8	1	2	4
2	2	1	1	4	2	2	2	1	1
3	1	2	1	2	4	3	1	2	1

As quantidades de recurso ofertada para os recursos 1, 2 e 3 são respectivamente 8, 10 e 8 unidades de recursos. Os valores agregados são os seguintes :

Tabela 4.16 - Valores agregados

	Tempos			Consumo recurso 1			Consumo recurso 2			Consumo recurso 3		
	1	2	3	1	2	3	1	2	3	1	2	3
A	8	9	5	5.75	5.00	2.80	3.25	1.67	2.80	3.25	2.33	2.80
B	6	7	7	4.50	1.57	3.14	4.00	2.14	3.14	1.50	4.14	2.14
C	7	7	12	5.29	2.57	5.25	1.57	4.71	2.75	2.14	2.57	3.50
D	4	7	12	1.50	3.14	7.00	2.50	1.57	1.75	1.25	3.14	2.50

Os valores obtidos para a classificação de recursos podem ser vistos nas tabelas 4.17 e 4.18.

Tabela 4.17 - Valores da média de consumo

Recurso	Média		
	M	M-1	M-2
1	11.68	7.92	3.96
2	7.96	5.31	2.65
3	7.82	5.21	2.61

Tabela 4.18 - Resultado da classificação dos recursos

Recurso	Oferta	T^r	% de coexistências			Classificação
			M	M-1	M-2	
1	8	49.38	87.50	52.78	0	crítico
2	10	23.50	20.83	0	0	não crítico
3	8	30.75	45.83	0	0	não crítico

Os resultados finais obtidos para este exemplo considerando o recurso como crítico são os seguintes:

Seqüência : C B D A
Makespan = 66 [u.t.]
Nós sondados = 14
Retrocessos = 0.

A tabela 4.19 apresenta os tempos de início e fim das operações para a construção da carta de Gantt, apresentada na figura 4.7.

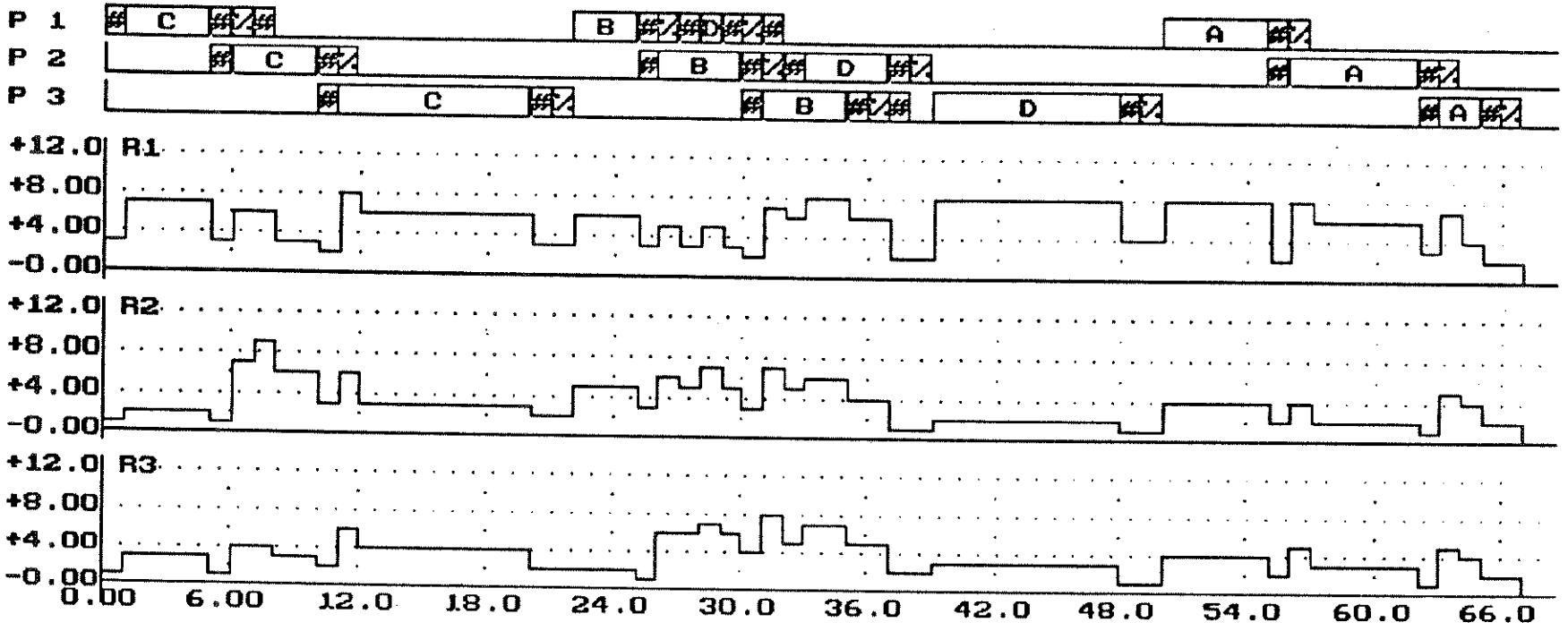
Este exemplo gerou como seqüência alternativa a seqüência A B C D cujo valor do makespan é igual a 68 [u.t.]. A figura 4.8 mostra o estado final da busca após a obtenção da seqüência final.

Tabela 4.19 - Tabela de tempos para a construção da carta de Gantt.

Tarefa	Processador	Tempos de início			
		Transferência	Processamento	Transferência	Preparação
C	1	0	1	5	6
	2	5	6	10	11
	3	10	11	20	11
B	1	7	22	25	26
	2	25	26	30	31
	3	30	31	35	36
D	1	27	28	29	30
	2	32	33	37	38
	3	37	39	48	49
A	1	31	50	55	56
	2	55	56	62	63
	3	62	63	65	66

Utilização de armazenagem intermediária

- produto D - início : 29 - fim : 33



LEGENDA:

# Transferência	Px - Processador x
% Preparação/Limpeza	Ry - Recurso y
X Processamento da tarefa x	eixo x - [u.t.]
# X # % Tarefa x	eixo y - [u.r.]

Figura 4.7 - Carta de Gantt da seqüência CBDA

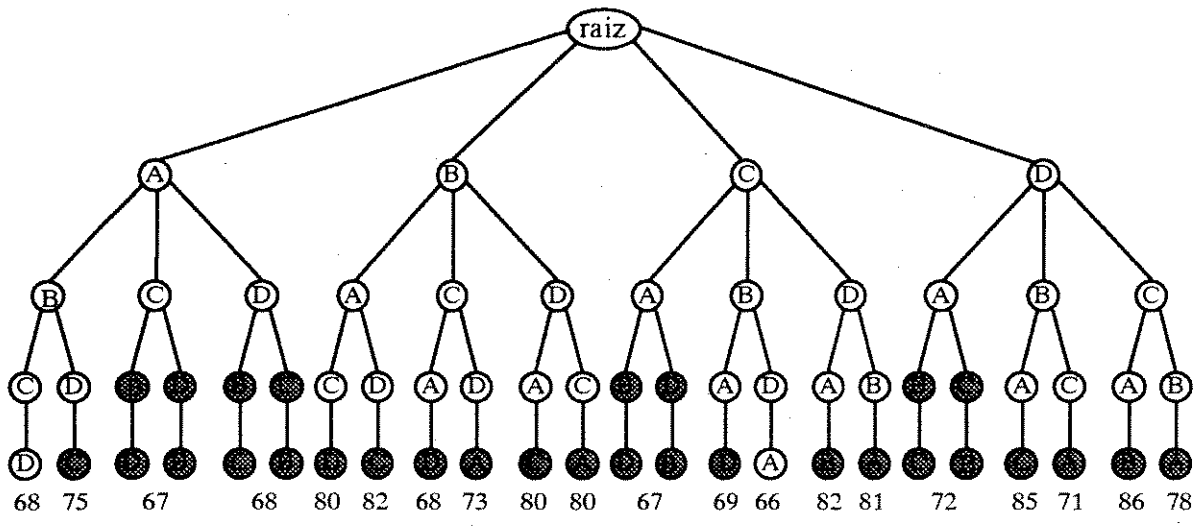


Figura 4.8 : Estado final da busca do seqüenciamento de C B D A.

4.3.3 Exemplo com consumos das operações de preparação e transferência maiores que os respectivos consumos das operações de processamento

Neste exemplo são mantidos os tempos das operações dos exemplos anteriores, são utilizados os valores dos consumos das operações de preparação e transferência do primeiro exemplo e o valor do consumo das operações de processamento são diminuídos pela metade. O valor da oferta é o mesmo do exemplo anterior, assim como o objetivo, que é o de verificar se a agregação de dados é adequada à situação proposta.

Os dados do exemplo são os seguintes :

Tabela 4.20 - Dados do exemplo

Tarefa A	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	5	6	2	1	1	1
Recursos									
1	8	6	4	4	3	2	8	6	4
2	4	2	4	3	1	2	4	2	4
3	4	3	4	2	2	2	4	3	4

Tarefa B	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	3	4	4	1	1	1
Recursos									
1	6	2	4	3	1	2	6	2	4
2	5	3	4	3	2	2	5	3	4
3	2	5	3	1	3	2	2	5	3

Tarefa C	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	4	4	9	1	1	1
Recursos									
1	7	3	6	4	2	2	7	3	6
2	2	6	3	1	3	2	2	6	3
3	3	3	4	2	2	2	3	3	4

Tarefa D	Preparação			Processamento			Transferência		
Processadores	1	2	3	1	2	3	1	2	3
Tempos	1	1	1	1	4	9	1	1	1
Recursos									
1	3	4	8	2	2	4	3	4	8
2	4	2	2	2	1	1	4	2	2
3	2	4	3	1	2	2	2	4	3

As quantidades de recurso ofertada para os recursos 1, 2 e 3 são respectivamente 8, 8 e 9 unidades de recursos. Os valores agregados são os seguintes :

Tabela 4.21 : Valores agregados

	Tempos			Consumo recurso 1			Consumo recurso 2			Consumo recurso 3		
	1	2	3	1	2	3	1	2	3	1	2	3
A	8	9	5	4.75	4.00	3.20	3.38	2.00	3.60	3.38	2.33	3.60
B	6	7	7	4.50	2.00	3.43	4.50	2.43	3.43	2.00	4.43	2.43
C	7	7	12	4.86	2.43	4.50	2.00	4.86	2.25	3.00	2.43	3.25
D	4	7	12	3.00	3.43	5.75	3.75	2.00	2.00	2.00	3.43	2.25

Os valores obtidos para a classificação de recursos podem ser vistos nas tabelas 4.22 e 4.23.

Tabela 4.22 - Valores das médias de consumo

Recurso	Média		
	M	M-1	M-2
1	10.89	8.17	5.44
2	7.68	5.76	3.84
3	7.57	5.68	3.79

Tabela 4.23 - Resultado da classificação dos recursos

Recurso	Oferta	T^r	% de coexistências			Classificação
			M	M-1	M-2	
1	8	42.75	83.33	36.11	0	crítico
2	10	21.60	12.50	0	0	não crítico
3	8	28.63	33.33	0	0	não crítico

Os resultados finais obtidos para este exemplo considerando o recurso 1 como crítico são os seguintes:

Seqüência : A B C D
Makespan = 61 [u.t.]
Nós sondados = 13
Retrocessos = 1

A tabela 4.24 mostra os tempos de início e fim das operações para a construção da carta de Gantt, apresentada na figura 4.10.

A figura 4.9 mostra o estado final da busca após a obtenção da seqüência final, e as seqüências parciais que eventualmente poderiam servir de alternativa à seqüência obtida.

Tabela 4.24 - Tabela de tempos para a construção da carta de Gantt.

Tarefa	Processador	Tempos de início			
		Transferência	Processamento	Transferência	Preparação
A	1	0	1	6	7
	2	6	8	14	15
	3	14	15	17	18
B	1	16	17	20	21
	2	20	21	25	26
	3	25	26	30	31
C	1	22	27	32	33
	2	32	34	38	39
	3	37	39	48	49
D	1	34	35	36	37
	2	40	41	45	46
	3	50	51	60	61

Utilização de armazenagem intermediária

- produto D - início : 36 - fim : 41
- produto D - início : 45 - fim : 51

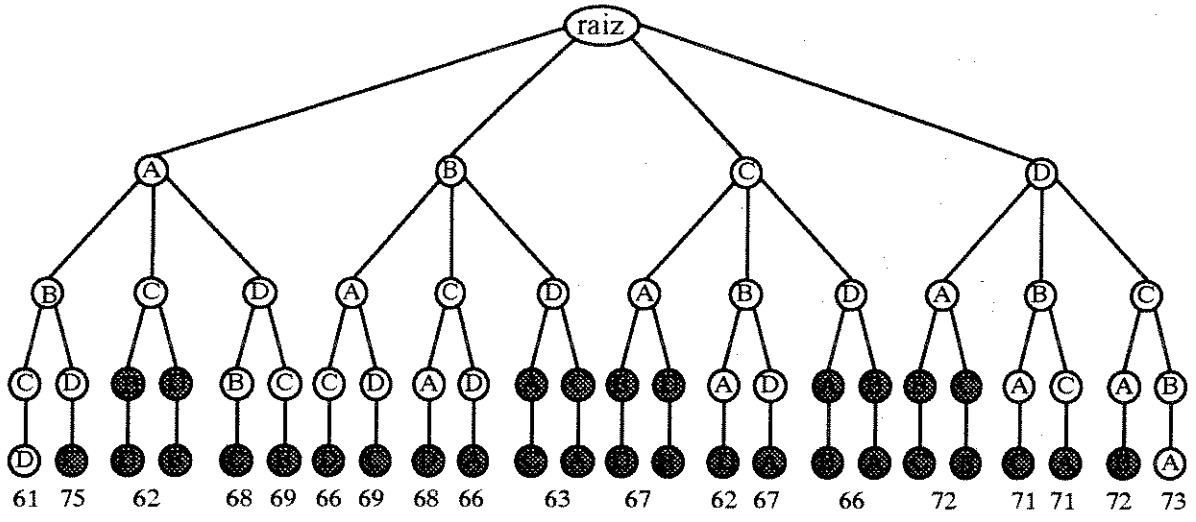
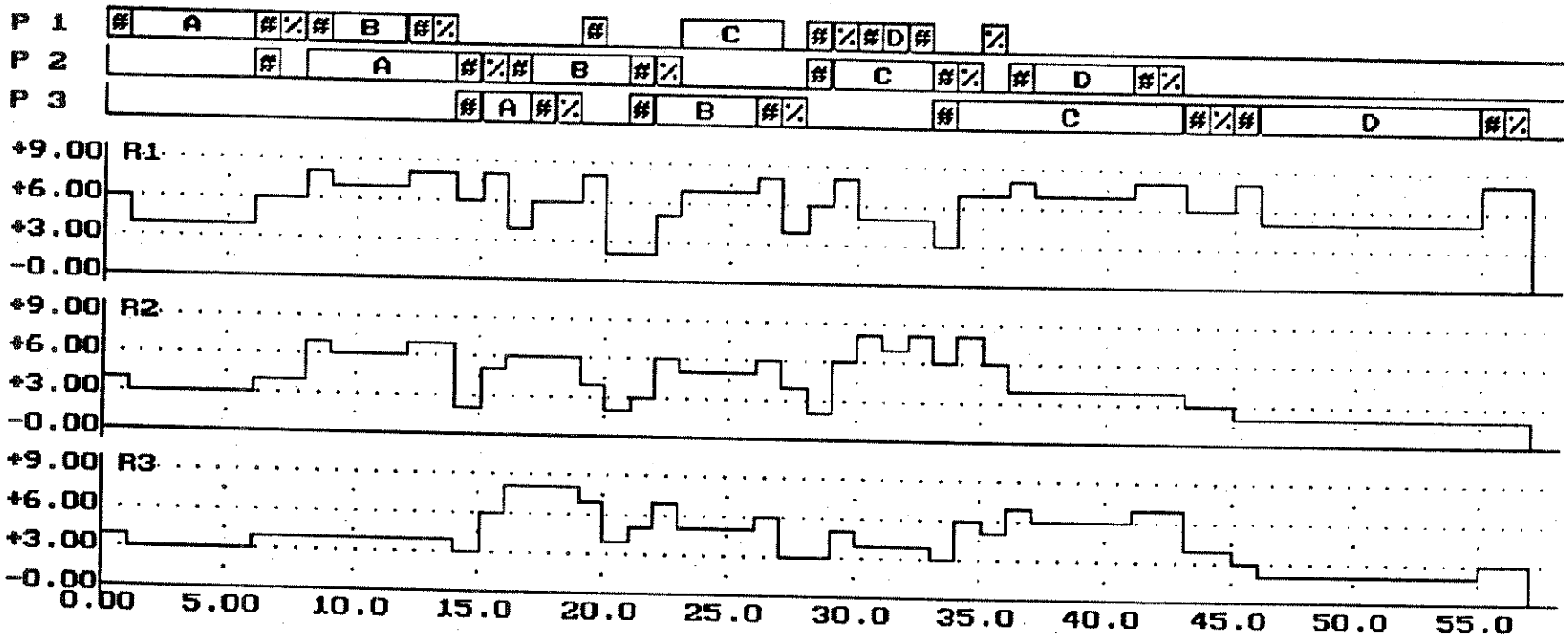


Figura 4.9 : Estado final da busca do seqüenciamento de A B C D.



LEGENDA:

# Transferência	Px - Processador x
% Preparação/Limpeza	Ry - Recurso y
X Processamento da tarefa x	eixo x - [u.t.]
# X # % Tarefa x	eixo y - [u.r.]

Figura 4.10 - Carta de Gantt da seqüência ABCD

4.3.4 Análise dos resultados apresentados nas seções 4.3.1, 4.3.2 e 4.3.3

Neste exemplos procurou-se explorar o comportamento do algoritmo onde, mantendo-se os tempos de fabricação, os valores das operações de preparação, processamento e transferência fossem variados considerando casos onde os consumos de cada uma das operações individualmente são considerados iguais, os consumos das operações de preparação e transferência são considerados menores ou maiores separadamente em relação a operação de processamento.

Analisando estes resultados é possível verificar que para estes casos o comportamento do algoritmo foi bastante semelhante em termos da quantidade de nós sondados e na quantidade de retrocessos para a obtenção da seqüência final, o que nos leva a concluir que a agregação de valores utilizando os valores médios de consumo das três operações e preservando a integral dos consumos é adequada. A seguir serão apresentados os resultados obtidos para um caso oito tarefas em uma planta com quatro processadores e três recursos de uso compartilhado.

4.3.5 Exemplo completo com utilização da "Beam-search"

Nesta seção são apresentados os resultados obtidos para o seqüenciamento de oito tarefas, numa planta com quatro processadores e três recursos de uso compartilhado. Nestes casos foram utilizados arbitrariamente tempos de preparação e transferência com valores inferiores, embora próximos, aos utilizados nas respectivas operações de processamento.

Os dados do exemplo são os seguintes:

Tabela 4.25 - Dados do exemplo

Tarefa A	Preparação				Processamento				Transferência			
Processadores	1	2	3	4	1	2	3	4	1	2	3	4
Tempos	1	1	1	1	5	6	2	2	1	1	1	1
Recursos												
1	6	5	4	4	8	6	4	5	6	5	4	4
2	3	2	3	4	4	2	4	5	3	2	3	4
3	2	3	3	2	3	4	3	2	2	3	3	2

Tarefa B	Preparação				Processamento				Transferência			
Processadores	1	2	3	4	1	2	3	4	1	2	3	4
Tempos	1	1	1	1	3	7	9	5	1	1	1	1
Recursos												
1	5	2	3	4	6	2	4	6	5	2	3	4
2	4	3	3	5	5	3	4	6	4	3	3	5
3	2	4	3	3	2	5	3	5	2	4	3	3

Tarefa C	Preparação				Processamento				Transferência			
Processadores	1	2	3	4	1	2	3	4	1	2	3	4
Tempos	1	1	1	1	4	3	2	4	1	1	1	1
Recursos												
1	5	3	4	5	7	3	6	5	5	3	4	5
2	4	2	4	4	4	2	4	5	4	2	4	4
3	3	2	3	3	3	3	4	3	3	2	3	3

Tarefa D	Preparação				Processamento				Transferência			
Processadores	1	2	3	4	1	2	3	4	1	2	3	4
Tempos	1	1	1	1	9	7	11	8	1	1	1	1
Recursos												
1	3	3	4	2	3	4	5	2	3	3	4	2
2	2	4	3	2	3	4	3	2	2	4	3	2
3	2	4	2	4	2	4	3	4	2	4	2	4

Tarefa E	Preparação				Processamento				Transferência			
Processadores	1	2	3	4	1	2	3	4	1	2	3	4
Tempos	1	1	1	1	3	4	9	5	1	1	1	1
Recursos												
1	3	3	4	4	3	4	5	4	3	3	4	4
2	3	1	2	3	3	1	2	4	3	1	2	3
3	2	1	2	2	3	1	2	4	2	1	2	2

Tarefa F	Preparação				Processamento				Transferência			
Processadores	1	2	3	4	1	2	3	4	1	2	3	4
Tempos	1	1	1	1	6	9	10	7	1	1	1	1
Recursos												
1	4	2	4	5	6	2	4	6	4	2	4	5
2	3	3	4	4	5	3	4	6	3	3	4	4
3	2	3	3	3	4	3	4	3	2	3	3	3

Tarefa G	Preparação				Processamento				Transferência			
Processadores	1	2	3	4	1	2	3	4	1	2	3	4
Tempos	1	1	1	1	5	4	4	5	1	1	1	1
Recursos												
1	5	3	4	5	7	3	6	5	5	3	4	5
2	3	2	2	3	4	2	2	4	3	2	2	3
3	2	3	3	3	2	4	3	4	2	3	3	3

Tarefa H	Preparação				Processamento				Transferência			
Processadores	1	2	3	4	1	2	3	4	1	2	3	4
Tempos	1	1	1	1	8	6	9	6	1	1	1	1
Recursos												
1	4	5	5	6	6	5	4	4	4	5	5	6
2	4	3	4	4	5	3	3	2	4	3	4	4
3	3	3	3	3	3	4	3	3	3	3	3	3

As quantidades de recurso ofertada para os recursos 1, 2 e 3 são respectivamente 15, 13 e 12 unidades de recursos. Os valores agregados para os tempos e os consumos são apresentados nas tabelas 4.26 e 4.27 respectivamente.

Tabela 4.26 - Valores dos tempos de processamento agregados

Tarefa Processador	A	B	C	D	E	F	G	H
1	8	6	7	12	6	9	8	11
2	9	10	6	10	7	12	7	9
3	5	12	5	14	12	13	7	12
4	7	8	7	11	8	10	8	9

Tabela 4.27 - Valores dos consumos agregados

Tarefa	A	B	C	D	E	F	G	H
Processador	Recursos 1							
1	7.25	5.50	6.14	3.00	3.00	5.33	6.25	5.46
2	5.67	2.00	3.00	3.70	3.57	2.00	3.00	5.00
3	4.00	3.75	4.80	4.79	4.75	4.00	5.14	4.25
4	4.57	5.25	5.00	2.00	4.00	5.70	5.00	4.67
Processador	Recurso 2							
1	3.63	4.50	4.00	2.75	3.00	4.33	3.63	4.73
2	2.00	3.00	2.00	4.00	1.00	3.00	2.00	3.00
3	3.40	3.75	4.00	3.00	2.00	4.00	2.00	3.25
4	4.57	5.63	4.57	2.00	3.63	5.40	3.63	2.67
Processador	Recurso 3							
1	2.63	2.00	3.00	2.00	2.50	3.33	2.00	3.00
2	3.67	4.70	2.50	4.00	1.00	3.00	3.57	3.67
3	3.00	3.00	3.40	2.79	2.00	3.77	3.00	3.00
4	2.00	4.25	3.00	4.00	3.25	3.00	3.63	3.00

Os valores obtidos para a classificação de recursos podem ser vistos nas tabelas 4.28 e 4.29 :

Tabela 4.28 - Valores das médias de consumo

Recurso	Média		
	M	M-1	M-2
1	17.69	11.80	5.90
2	13.51	9.00	4.50
3	12.08	8.06	4.03

Tabela 4.29 - Resultado da classificação de recursos

Recurso	Oferta	T^r	% de coexistências			Classificação
			M	M-1	M-2	
1	15	82.73	86.37	21.50	0	crítico
2	13	73.77	63.81	0	0	não crítico
3	12	72.50	53.81	0	0	não crítico

A seguir serão apresentados os resultados finais obtidos para este exemplo:

- Seqüência final = B A E D G C F H

- Makespan estimado pelo T^r = 83 [u.t.]
- Makespan estimado pela regra SMBB = 86 [u.t.]
- Makespan obtido para a seqüência final = 114 [u.t.]
- Total de nós existentes na árvore = 109600
- Quantidade de nós sondados = 2615
- Retrocessos entre a factibilização e o B&B = 6

A tabela 4.30 apresenta os resultados de simulação para diferentes situações utilizando-se a "Beam-search".

Tabela 4.30 - Resultados de simulação utilizando "Beam-search"

Valor Beam	Nós sondados	Seqüência	Makespan	Retrocessos
7	1872	BHFGDECA	114	5
6	1329	BHFGDECA	114	5
5	781	BHFGDECA	114	5
4	349	BHFGDECA	114	5
3	133	CEDGBHFA	115	3
2	13	CEBGDHFA	117	0
1	7	EGBCDHFA	118	0

Comentários:

Este exemplo tem como objetivo demonstrar o desempenho do algoritmo como um todo. De um total de 109600 nós existentes na árvore de busca o algoritmo sondou apenas 2615 nós o que corresponde a 2.39% do total, este dado mostra que a função de estimativa do "lower-bound" é bastante eficiente. A quantidade de retrocessos 6, que significa que apenas seis seqüências foram analisadas detalhadamente, quando comparada à quantidade de ramos percorridos na árvore 2261 mostra que a proposta de classificação dos recursos e de agregação de dados é bastante adequada à forma pela qual o problema é tratado.

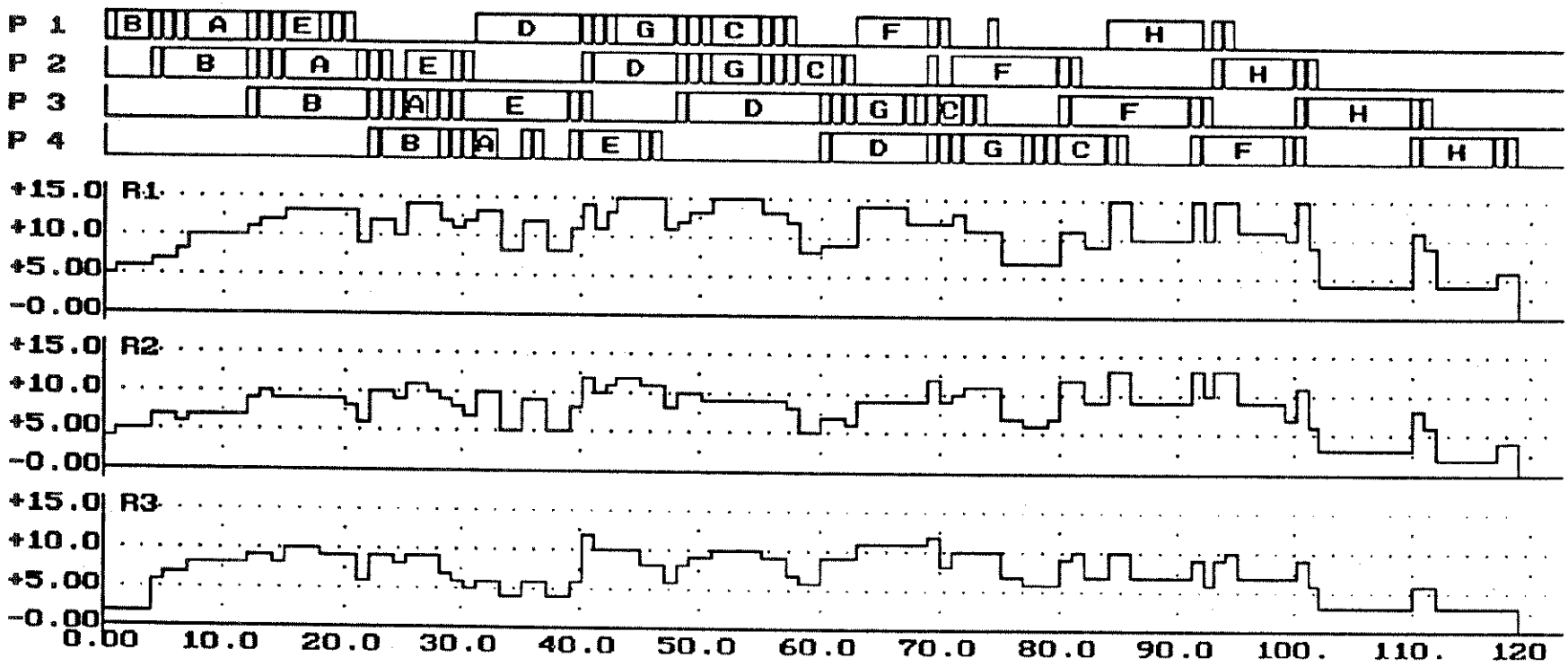
A utilização de "Beam-search" mostra que até valores de "beam" igual a 4 foi possível encontrar soluções com o mesmo valor de "makespan" obtido para a seqüência final sem a utilização da mesma, embora não seja possível afirmar que isto sempre ocorrerá. A utilização da "Beam-search" é uma alternativa interessante a ser utilizada em problemas de grande dimensão, onde o tempo necessário para a obtenção da seqüência final pode ser proibitivo para o "Branch-and-Bound".

As tabela 4.31 e 4.32 mostram, respectivamente, a utilização da armazenagem intermediária e os tempos de início e fim das operações para a construção da carta de Gantt apresentada na figura 4.11.

Tabela 4.31 - Utilização da armazenagem intermediária

Produto	início	fim
A	12	15
	21	25
	27	31
E	18	24
G	48	51
	55	63
	67	72
C	55	58
	61	70
	72	80

	Tabela 4.32 - Tempos de início para a construção da carta de Gantt				
Tarefa	Processador	Transferência	Processamento	Transferência	Preparação
B	1	0	1	4	5
	2	4	5	12	13
	3	12	13	22	23
	4	12	23	28	29
A	1	6	7	12	13
	2	14	15	21	22
	3	24	25	27	28
	4	30	31	35	36
E	1	14	15	18	19
	2	23	25	29	30
	3	29	30	39	40
	4	39	40	45	46
D	1	20	31	40	41
	2	40	41	48	49
	3	48	49	60	61
	4	60	61	69	70
G	1	42	43	48	49
	2	50	51	55	56
	3	62	63	67	68
	4	71	72	77	78
C	1	50	51	55	56
	2	57	58	61	62
	3	69	70	72	73
	4	79	80	84	85
F	1	57	63	69	70
	2	69	71	80	81
	3	80	81	91	92
	4	91	92	99	100
H	1	74	84	93	95
	2	93	94	100	101
	3	100	101	110	111
	4	110	111	117	118



LEGENDA:

x Tarefa x

Px - Processador x

Ry - Recurso y

eixo x - [u.t.]

eixo y - [u.r.]

Figura 4.11 - Carta de Gantt da seqüência BAEDGCFH

4.4 Conclusão

Neste capítulo foi analisado o desempenho do algoritmo proposto utilizando duas situações distintas. Na primeira utilizou-se o mesmo modelo (agregado) na fase de geração de seqüências e na fase de factibilização. O objetivo foi analisar a metodologia proposta para classificação de recursos. Observou-se uma melhora no desempenho quando são eliminados os recursos claramente não críticos e um desempenho ruim quando os recursos evidenciados como mais críticos não são incluídos na fase de geração de seqüências. No que se refere aos recursos com criticalidade não bem definida, seja em termos de T^r ou de porcentagem de coexistências acima da oferta, a proposta é de utilizar uma estratégia conservadora do tipo "na dúvida não se elimina o recurso da fase de geração de seqüências". Isto porque os exemplos mostram que pode ocorrer uma piora no desempenho se o processo de redução do número de recursos naquela fase é exagerado.

Na segunda situação, o algoritmo proposto foi testado com exemplos de simulação considerando as operações de preparação, processamento e transferência com consumos específicos de recursos. Neste caso a fase de geração de seqüências utiliza um modelo agregado e a fase de factibilização um modelo detalhado. Os resultados mostram que a abordagem proposta é bastante adequada quando se pretende ter um seqüenciamento com este nível de detalhe. Isto se deve à abordagem multi nível e à decomposição do problema, as quais possibilitam trabalhar com os diferentes modelos.

Como na literatura, até onde se pesquisou, não existem propostas semelhantes na qual o seqüenciamento é tratado com o mesmo nível de detalhe em relação as operações e com restrição rígida na oferta de recursos não foi possível fazer um estudo comparativo da abordagem proposta.

Embora não tenha sido feita uma análise estatística para verificar o desempenho do algoritmo, chegou-se a conclusão que o desempenho do algoritmo é bom, tendo apresentado em geral uma porcentagem pequena de nós sondados em relação à quantidade total existente, gastando para isto um tempo bastante razoável, mesmo não utilizando computadores sofisticados.

A utilização de heurísticas do tipo "Beam-search" e a consideração de tarefas alocadas externamente, em conjunto com "upper-bounds" são estratégias importantes na redução do espaço de busca.

Em relação aos dois tipos critérios utilizados na classificação de recursos, é possível comentar o seguinte:

- o critério baseado na demanda total de recursos além de ser facilmente calculado se mostra ser bastante eficaz na definição dos recursos críticos; e
- o critério baseado na coexistência de operações traz para a classificação dos recursos uma informação complementar à respeito da limitação que certos recursos impõem em relação ao número de operações que podem coexistir. Se por um lado esta é uma informação importante, por outro, necessita de um esforço computacional superior ao anterior.

Capítulo 5 - Conclusões

Neste trabalho foi proposto um algoritmo de seqüenciamento para o problema de flowshop com restrições na oferta de recursos de uso compartilhado e com tarefas alocadas externamente. Foi considerado no problema um modelo detalhado das operações de processamento de uma tarefa.

A estratégia proposta utiliza uma abordagem multi nível, na qual o problema é subdividido em três níveis: pré-seqüenciamento, seqüenciamento e pós-seqüenciamento. No nível de seqüenciamento o problema é subdividido em dois outros níveis. O primeiro é o responsável pela geração de seqüências através de um algoritmo de "Branch and Bound" utilizando um modelo agregado da planta considerando apenas os recursos críticos. O segundo, um nível de factibilização utilizando um modelo detalhado, no qual se consideram todos os recursos.

Este tipo de abordagem se mostrou adequada para tratar problemas mais complexos através da decomposição do problema. A geração de seqüências utilizando o BAB é justamente a parte do algoritmo mais crítica, pois o BAB na busca da solução ótima explora um espaço de busca da solução de uma dimensão significativa. Desta forma a utilização de um modelo agregado do problema e a consideração exclusiva dos recursos críticos no nível de geração de seqüências torna o problema mais simples que o problema global que está sendo tratado, possibilitando assim o tratamento de problemas mais complexos.

Esta estratégia se completa com os demais níveis onde os dados manipulados pelo BAB são tratados. O nível de pré-seqüenciamento é responsável pela classificação dos recursos e pela geração do modelo agregado. Por ser uma fase anterior ao BAB, e portanto não estar influenciado pela dinâmica do mesmo, pode então executar algoritmos sofisticados que contribuem para a solução mais eficiente do BAB. A fase de factibilização complementa o BAB fazendo uma análise detalhada da solução para o problema global.

A incorporação de heurísticas no algoritmo de BAB e a consideração de tarefas alocadas externamente é certamente um passo seguro na diminuição do espaço de busca de solução, pois aumentam a aplicabilidade do algoritmo.

Os resultados apresentados no capítulo 4 mostram que a proposta de classificação de recursos é sem dúvida uma estratégia interessante pois melhora o desempenho do algoritmo sem comprometer a obtenção da solução ótima. Este tipo de estratégia parece

ser bastante natural, pois na prática o planejador se preocupa preferencialmente com os recursos que são mais críticos, pois ele sabe que os demais não tem grande influência na solução do problema. Os resultados mostraram também que o algoritmo tem um bom desempenho para resolver problemas onde se considera modelos mais detalhados da planta e que a abordagem multi nível é adequada para resolver problemas de seqüenciamento.

A classificação de recursos, a qual possibilita separar recursos em críticos e não críticos, contribui para melhorar o desempenho do algoritmo de seqüenciamento. No entanto, como mostram os exemplos do capítulo 4, a separação dos recursos nestes dois conjuntos não é tarefa trivial. Existem recursos que por estarem nos extremos são facilmente classificados, mas existem outros que por se encontrarem numa região de transição a sua classificação passa a ser algo um tanto duvidoso. Os diversos casos testados mostraram que considerar um recurso não crítico como crítico leva a um desempenho pior do que se considerarmos apenas os críticos, mas o desempenho é sempre melhor ou no pior caso igual ao desempenho da abordagem global onde todos os recursos são considerados críticos. Já no outro caso, i.e., considerar um recurso crítico como não crítico, pode levar a um desempenho nada favorável. Portanto, o que se propõe é uma estratégia de eliminação de recursos claramente não críticos, ou seja, na dúvida da criticalidade de um recurso considera-se como crítico, esta abordagem se mostrou ser a mais robusta.

Esta situação se explica porque os indicadores de criticalidade dos recursos, calculados antes do seqüenciamento, são abstrações um tanto grosseiras para representar os problemas específicos de alocação que os recursos impõem durante o seqüenciamento. Estes indicadores poderiam ser complementados com outros obtidos a partir de um procedimento de reconhecimento de padrões atuando sobre um algoritmo de busca em árvore trabalhando com todos os recursos, excluídos os claramente não críticos. Os padrões a serem reconhecidos neste caso seriam as características das tarefas/recursos que fazem com que alguns ramos da árvore não sejam pesquisados porque não conduzem à solução ótima. Esta estratégia se aplicaria a problemas específicos e bastaria executá-lo apenas uma vez enquanto não mudarem as características do problema, neste caso da planta e dos produtos. Note que as quantidades de produto, datas de início, prazos de entrega podem e devem (em função das necessidades do cliente) ser alterados.

A utilização da "Beam-search" tal qual apresentada no capítulo 4, embora seja interessante pois contribui para a redução do espaço de busca, poderia ser melhorada através de uma estratégia na qual em vez de considerar a eliminação de nós filhos através de valores fixos, utilizar valores variáveis que poderiam ser definidos em função da

profundidade da busca e da comparação da importância relativa dos nós filhos definidos através de conhecimento específicos do problema. A estratégia consiste em eliminar ramos da árvore com pouco potencial de condução à solução ótima.

Neste trabalho procurou-se dar uma contribuição para o problema de seqüenciamento na Indústria de Processos Químicos Descontínuos. O projeto, planejamento da produção e seqüenciamento deste tipo de plantas químicas é uma área de trabalho recente onde as primeiras abordagens, utilizando técnicas de Pesquisa Operacional, aparecem na literatura a partir de 1979 (Grossmann & Sargent (1979), Mauderli & Rippin (1980)). A aplicação de técnicas de Inteligência Artificial é mais recente e em particular as técnicas de busca orientada pelas restrições ("Constraint Based Search") que parecem mais adequadas começaram a ser aplicadas ao problema de seqüenciamento em 1983 (Fox). A partir da experiência adquirida no desenvolvimento deste trabalho de pesquisa pode-se propor trabalhos futuros considerando para tanto duas perspectivas:

1)Melhorias específicas da abordagem proposta:

- desenvolver outros indicadores de criticalidade dos recursos;
- adicionar novas heurísticas ao processo de busca;
- considerar também critérios de custo baseados nos custos de produção e no lucro;
- considerar outros tipos de políticas de armazenagem, como as do tipo NW ("No-Wait"), FIS ("Finite Intermediate Storage"), etc.

2) Extensão da proposta em termos genéricos:

- considerar o problema de plantas químicas multi-propósito;
- considerar o problema de seqüenciamento dentro de uma ótica do planejamento global, considerando as decisões de médio, longo e curto prazo;
- considerar o problema do seqüenciamento reativo.

Referências

Atabakhsh H., "A Survey of Constraint Based Scheduling Systems Using an Artificial Intelligence Approach", *Artificial Intelligence in Engeneering*, vol.6, n.2, 1991.

Axsäter S., "Aggregation of Product Data for Hierarchical Production Planning", *Operation Research,* vol. 29, July/Aug. 1981, pp. 744-756.

Baker K.R., *Introduction to Sequencing and Scheduling*, John Wiley, 1974.

Bensana E., Bel G., e Dubois D., "OPAL: A Multiknowledge-Based System for Industrial Job-Shop Scheduling", *International Journal of Production Research*, vol.26, n.5, 1988.

Bensana E., "Melting OR and AI Techniques: Toward a Short Term Scheduling Environment", International Conference on Manufacturing Systems and Environment : Looking Toward the 21th Century, Tokio, May 1990, pp273-312.

Booch G., "On the Concepts of Object-Oriented Design", Chapter 6 - Modern Software Engineering Foundation and Current Perspectives, Van Norstrand Reinhold - NY, 1991.

Boskma K., "Aggregation and the Design of Models for Medium-Term Planning of Production", *European Journal of Operational Research*, 10, 1982, pp 244-249.

Campos M.F.D., "Seqüenciamento e alocação de operações em flow-shops com restrições sobre os recursos compartilhados e sobre o prazo de entrega das tarefas: uma abordagem de busca orientada por restrições", Tese de Mestrado, Unicamp, agosto de 1993.

Campos M.F.D., Latre L.G., Rodrigues M.T.M., e Passos C.A.S.,"Seqüenciamento de flow-shops com restrições sobre os instantes de alocação das tarefas e sobre os recursos compartilhados", XXV SBPO, Campinas, novembro 1993.

Carlier J., Chrétienne P., "Un Domaine très Ouvert : Les Problémes d'Ordonnancement", *RAIRO* vol. 16, no. 3, août 1982, p. 175 à 217.

Coffman Jr. F.G., *Computer and Job-Shop Scheduling Theory,* John Wiley, New York - 1976.

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

Dannenbring D.G., "An Evaluation of Flow Shop Sequencing Heuristics", *Management Science*, 23(11), 1977, pp. 1174-1182.

Dauzere-Perez S., "Planification et Ordonnancement de la Production : Une Approche Integrée Coeherente", Thèse de Doctorat, Université Paul Sabatier, Toulouse, 1992.

Dechter R., "Enhancement Schemes for Constraint Processing: Backjumping, Lerning, and Cutset Decomposition", *Artifical Intelligence*, 41, 1990 pp 273-312.

Dempster M.A.H., e outros, "Analytical Evaluation of Hierarchical Planning Systems", *Operations Research*, Vol. 29, no. 4, July-August 1981, pp. 707-716.

Dudek R.A., Pawalkar S.S., e Smith M.L., "The lessons of flowshop scheduling research", *Operations Research*, vol.40, n.1, 1992.

Egli U.M., Rippin D.W.T., "Short-Term Scheduling for Multiproduct Batch Chemical Plants", *Comp. Chem. Engng*, 10(4), 1986, pp. 303.

Erschler J., e Esquirol P., "Decision-Aid in Job-Shop Scheduling: a Knowledge-Based Approach", *IEEE*, 1986.

Erschler J., "Analyse Sous Contraintes et Aide `a la Décision Pour Certains Problèmes d'Ordonnancement", Thèse de Doctorat, Université Paul Sabatier, Toulouse, 1976.

Esquirol P., "Règle et Processus d'Inference pour l'Aide a l'Ordonnancement de Tâches en Presence de Contraintes", Thèse de Doctorat, Université Paul Sabatier, Toulouse, 1987.

Fox M.S., "Constraint Directed Search: A Case Study of Job-Shop Scheduling", Ph.D. Thesis, Carnegie-Mellon University, 1983.

French S., *Sequencing and Scheduling: An Introduction to the Matematics of the Job-Shop*, Ellis Horwood- 1982.

Garey M.R., Johnson D.S., *Computers and Intractability; a Guide of NP-Completeness*, New York, W. H. Freemon and Company, 1979, 340 p.

Graves S.C., "A review of production scheduling", *Operations Research*, vol.29, n.4, 1981.

Grossmann I.E., Sargent R.W.H.,"Optimum Design of Multipurpose Chemical Plants", *Ind. Eng.. Chem. Process. Dev.*, Vol. 18 n.2, 1979, pp 343-348.

Horowitz E., Sahni S., *Fundamental of Computer Algorithms*, Computer Science Press, Rockville, MD, 1978.

Hu D., *Object-Oriented Environment in C++: A User-Friendly Interface*, MIS:Press, 1990.

Imbert S., "Interaction Entre Deux Niveaux de Decision en Planification de la Production", Thèse de Doctorat, Université Paul Sabatier, Toulouse, 1986.

Janiak A., "General Flow-shop scheduling with Resource Constraint", Int. J. Prod. Res., vol. 26(6), 1988, pp. 1089-1103.

Keng N.P., Yun D.Y.Y., e Rossi M., "Interaction-sensitive planning system for job-shop scheduling", *Expert Systems and Intelligent Manufacturing*, 1988, pp. 57-69.

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

Lassere J.B., Martin J.P., e Roubellat F., "Aggregate Model and Decomposition Method for Midi-Term Production Planning", *Int. J. Prod. Res.*, vol 21, no. 6, 1983, pp. 835-843.

Le Pape C., "Constraint Propagation in Planning and Scheduling", Robotics Laboratory Department of Computer Science - Stanford University - Jan. 1991.

Lopez P., "Approche Energetique pour l'Ordonnancement de Taches Sous Contraintes de Temps et de Ressources, Thèse de Doctorat, Université Paul Sabatier, Toulouse, 1991.

Mauderli A., Rippin D.W.T.,"Scheduling Production in Multi-Puprose Batch Plants: The Batchman Program", *AIChE*, CEP April 1980, pp 37-45.

Nawaz M., Enscore E., Ham I., "A Heuristic Algorithm for the M-Machine, N job Sequencing Problem", *The Int. Jl. of Mgmt. Sci.*, 11(1), 1983, pp. 93.

Nilsson N., *Problem Solving in Artificial Intelligence*, McGraw Hill, 1971.

Papadimitriou C.H., Steiglitz K., *Combinatorial Otimization: Algorithms and Complexity*, Prentice-Hall, Inc., 1982.

Parrello B.D., Kabat W.C., Wos L.,"Job-Shop Scheduling Using Automated Reasoning: A Case Study of the Car-Sequencing Problem", Journal of Automated Reasoning, Vol 2, no. 1 , 1986, pp 1-42.

Pohl I.., C++ for Pascal Programmers, The Benjamim/Cummings Publishing Company, Inc., 1991.

Potts C.N., e Van Wassenhove L.N., "Integrating Scheduling with Batching and Lot-Sizing: a Review of Algorithms and Complexity", *J. Opl Res. Soc.*, Vol. 43, n.5, 1992, pp 395-406.

Rich E., *Artificial Intelligence*, McGraw-Hill Book Company, 1983.

Rickel J., "Issues in the Design of Scheduling Systems", *Expert Systems and Intelligent Manufacturing*, Elsevier Science Publishing Co., Inc. 1988 - pp. 70-89.

Rodammer F.A., e White K.P., "A Recent Survey of Production Scheduling", *IEEE Trans. Systems, Man, and Cybernetics*, vol.18, No.6, pp.841-851, 1988.

Rodrigues M.T.M., "Seqüenciamento e Alocação de Operações em Flow Shops na Indústria Química com Restrições sobre os recursos Compartilhados. Uma abordagem de Busca Orientada por Restrições", Tese de Doutorado, UNICAMP, 1992.

Rodrigues M.T.M., Passos C.A.S., Latre L.G., e Campos M.F., "Flow Shop Scheduling with Shared Resources : A Branch and Bound Approach", *ISRAM'92* - Santa Fé, New Mexico, EUA, Nov.92.

Rodrigues M.T.M., Passos C.A.S., Latre L.G., e Campos M.F., "Flow Shop Scheduling: A Resource Based Branch and Bound Perspective", *12th IFAC World Congress*, Sydney, Australia, july 1993.

Rodrigues M.T.M., Passos C.A.S., Latre L.G., e Campos M.F., "A Resource Based Branch and Bound Algorithm for Flow Shop Scheduling", Anais do I Simpósio Brasileiro de Automação Inteligente, UNESP - Rio Claro, setembro de 1993.

Sacerdoti E.D., "Planning in a Hierarchy of Abstrations Spaces", *Artificial Intelligence*, vol.5, 1974.

Sadeh N., e Fox M.S., "CORTES: An Exploration Into Micro-Opportunistic Job-Shop Scheduling", *AAAI-Sigman - Workshop on Manufacturing Scheduling / IJCAI - 89/23, 1989.*

Shlaer S., Mellor S.J., *Object-Oriented Systems Analysis: Modeling the World in Data*, Yordon Press, 1988.

Smith S.F., e Ow P.S., "The Use of Multiple Problem Decompositions in Time Constrained Planning Tasks", *Proceedings of the 9th. International Joint Conference on Artificial Intelligencel*, 1985, pp.1013-1015.

Smith S.F., Ow P.S., Potvin J.Y., Muscettola N., e Matthys D.C., "An integrated framework for generating and revising factory schedules", *Journal of the Operational Research Society*, vol.41, n.6, 1990.

Stefik M., Bobrow D.G., "Object-Oriented Programming: Themes and Variations", *The AI Magazine*, 1986, pp. 40-61.

Steffen M.S., e Greene T.J., "An Application of Hierarchical Planning and Constraint-Directed Search to Scheduling Parallel Processors", *IEEE*, 1986, pp. 910.

Stroustrup B., *The C++ Programming Language*, Addison Wesley - 1986

Sycara K., Roth S., Sadeh N.,e Fox M.S., "Distributed Constrained Heuristic Search", *IEEE Trans. systems, man, and cybernetics*, vol.21, n.6, 1991.

Sycara K., Roth S., Sadeh N., e Fox M.S., "Resource Allocation in Distributed Factory Scheduling", *IEEE Expert*, february 1991, pp. 29.

Wilkins D.E., "Hierarchical Planning: Definition and Implementation", *Advances in Artificial Intelligence-II*, B.D.Boulay, D.Hogg and L.Steels(Editors), Elsevier Science Publishers B.V., North-Holland, 1987.

Wilkins D.E., "Domain-independent Planning: Representation and Plan Generation", *Artificial Intelligence*,, Elsevier Science Publishers B.V., North-Holland, 1984, pp 269-302.

Zorzo C.A., Cury J.E.R., "Uma Estratégia para Escalonamento de Tarefas em Tempo Real para Sistemas Flexíveis de Manufatura", *9º CBA - SBA*, UFES - Vitória/ES, 1992, pp. 1175-1180.

Anexo A - Aspectos da implementação orientada ao objeto

Na abordagem proposta o problema é decomposto em níveis e subníveis com o objetivo de simplificar o problema no nível mais crítico em termos de processamento, no caso o nível onde é executado o "Branch-And-Bound", sem no entanto desprezar detalhes considerados importantes na solução do problema. O objetivo desta abordagem é poder utilizar modelos com diferentes níveis de abstração a cada nível, além de facilitar a utilização de diferentes algoritmos. Isto fez com que se procurasse uma técnica de programação na qual estas características do problema fossem tratadas de uma forma direta e natural. Uma técnica que se mostrou adequada é a orientada ao objeto. Esta metodologia possui como características principais a existência de herança entre as classes, o encapsulamento de dados e polimorfismos.

A proposta deste anexo não é a de introduzir o conceito da programação orientada ao objeto ("Object Oriented Programming"- OOP), mas sim apresentar os principais aspectos relacionados com o projeto e a implementação da abordagem proposta. Maiores detalhes podem ser obtidos em Stroustrup(1986), Stefik (1986), Shlaer (1988), Hu (1990) e Booch (1991). A seguir será apresentada uma definição de objetos que é o conceito básico da OOP, uma breve descrição da metodologia do projeto, e uma discussão sobre a linguagem utilizada.

A.1 O significado de um objeto (Booch (1991))

O conceito de objeto é central para qualquer tipo de sistema orientado ao objeto. Um objeto é definido como sendo uma entidade com as seguintes características:

- tem estado;
- é caracterizado pelas ações que sofre e que requer de outros objetos;
- é uma instância de alguma classe (possivelmente anônima);
- é denotado por um nome;
- tem uma visibilidade restrita de e por outros objetos; e
- pode ser "visto" ou por sua especificação ou por sua implementação.

No projeto de um sistema orientado ao objeto, a primeira preocupação é com a visão externa dos objetos. A visão externa de um objeto ou classe é a sua especificação. A especificação captura toda a semântica estática e dinâmica do objeto. Além da visão externa, existe a visão interna, que é a sua implementação, e a qual não é visível externamente.

Objetos não existem isoladamente, desta forma, é importante considerar o relacionamento entre os mesmos. A figura A.1 representa a estrutura de um sistema decomposto segundo a técnica orientada ao objeto. Note que a unidade fundamental da decomposição é o objeto, e que a topologia do sistema é de um DAG ("Direct Acyclic Graph"), com os dados distribuídos pelo sistema e encapsulados dentro dos objetos.

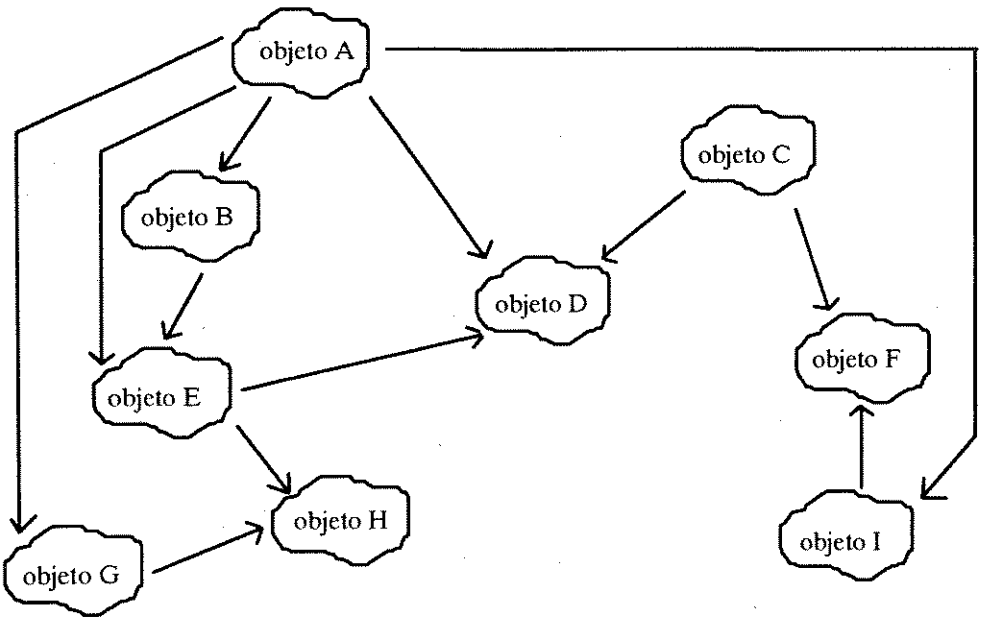


Figura A.1 : Topologia de um sistema usando a técnica orientada ao objetos

A.2 Projeto orientado ao objeto

Segundo Booch (1991), a programação orientada ao objeto não provê uma abordagem sistemática para a decomposição de sistemas complexos e, desta forma, não constitui um método de projeto. Entretanto, como existem fundamentos teóricos para a técnica orientada ao objeto, Booch desenvolveu uma abordagem chamada de "Projeto Orientado ao Objeto", que será apresentada resumidamente a seguir.

Para qualquer método ser considerado completo ele deve contemplar o seguinte :

- decomposição física e lógica; e
- comportamento estático e dinâmico.

Além disso, o método deve contemplar as duas dimensões da estrutura do sistema, que são peculiares aos sistemas orientados ao objeto, chamados de :

- hierarquia pai-filho; e
- decomposição de objetos e classes.

Um programa que implementa um modelo da realidade pode ser visto como sendo um conjunto de objetos que interagem um com os outros. O projeto orientado ao objeto é baseado nesta visão. Os principais passos de um projeto baseado na OOP são os seguintes:

1. Identificar os objetos e seus atributos;
2. Identificar as operações sofridas por e requeridas de cada objeto;
3. Estabelecer a visibilidade de cada objeto em relação aos demais;
4. Estabelecer as interfaces entre os objetos; e
5. Implementar os objetos.

A.3 Etapas do projeto

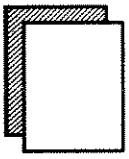
A seguir são apresentadas as principais etapas do projeto.

A.3.1 Definição dos objetos

Uma vez definidos os principais aspectos relacionados com o problema, é possível elaborar um Diagrama de Fluxo de Dados - DFD, o qual permite representar o comportamento do sistema e capturar o modelo do espaço do problema (Booch (1991)).

A utilização de um DFD, embora não seja a única forma de fazê-lo, é um passo preliminar no projeto de um programa orientado ao objeto. Numa programação convencional, utilizando um método funcional, ele serviria para definir as funções do sistema e seria continuado através da utilização de diagramas estruturados usando por exemplo um Diagrama Hierárquico de Funções (DHF) ou um "Structure Chart". Já no projeto orientado ao objeto, ele serve para extrair os objetos. O processo de obtenção destes objetos será comentado nos itens a seguir. Note que na OOP a utilização dos termos objetos e classes muitas vezes se confundem, pois um objeto está intimamente ligado à definição de uma classe, embora sejam conceitualmente diferentes.

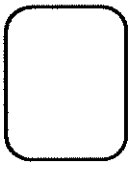
Na figura A.2 é apresentado o DFD, em nível macroscópico, da abordagem proposta, onde são apresentadas as principais funções existentes no projeto com as respectivas origem e destino dos dados. Este diagrama utiliza a notação de Gane e Sarson (Booch (1991)), onde os elementos representam o seguinte :



quadrado duplo - representa a origem ou destino dos dados

retângulo com fim aberto - representa a armazenagem dos dados

linha direta - representa o fluxo dos dados



retângulo arredondado - representa um processo que transforma os dados

Por exemplo, como mostrado na figura A.2, a classificação dos recursos é realizada a partir dos dados referentes aos tempos das operações, as tarefas a serem seqüenciadas e aos recursos, sendo o resultado final, a própria classificação. Da mesma forma a alocação externa de tarefas é realizada a partir dos dados das tarefas a serem seqüenciadas e da interação com o planejador, o resultado é a definição das tarefas alocadas e suas respectivas janelas.

Este tipo de DFD pode ser explodido em vários níveis, cada nível contendo um maior detalhamento de seus elementos, até que se tenha uma visão clara de todos os objetos/classes existentes.

A partir da análise do DFD da figura A.2 é possível identificar os principais objetos que farão parte do projeto, são eles : Tempos, Tarefas, Recursos, Classificação, Seqüenciamento e Pós-seqüenciamento. Estes são os nomes escolhidos para representar os objetos, e não seguem nenhuma regra pré-estabelecida.

Estes objetos, apesar de alguns serem considerados objetos abstratos, são facilmente identificáveis a partir do DFD. Dentre eles estão os objetos : Tempo, Tarefas, Recursos e Classificação. Já os objetos Seqüenciamento e Pós-seqüenciamento parecem menos evidentes, mas na realidade por possuírem todas as características discutidas anteriormente a sua definição acaba sendo relativamente simples. O objeto seqüenciamento, por exemplo, contempla além dos algoritmos, a lista de seqüências e as curvas de utilização dos recursos. Desta forma, é fácil verificar que o mesmo tem estado, determinar as ações que sofre e requer de outros objetos, bem como as demais características.

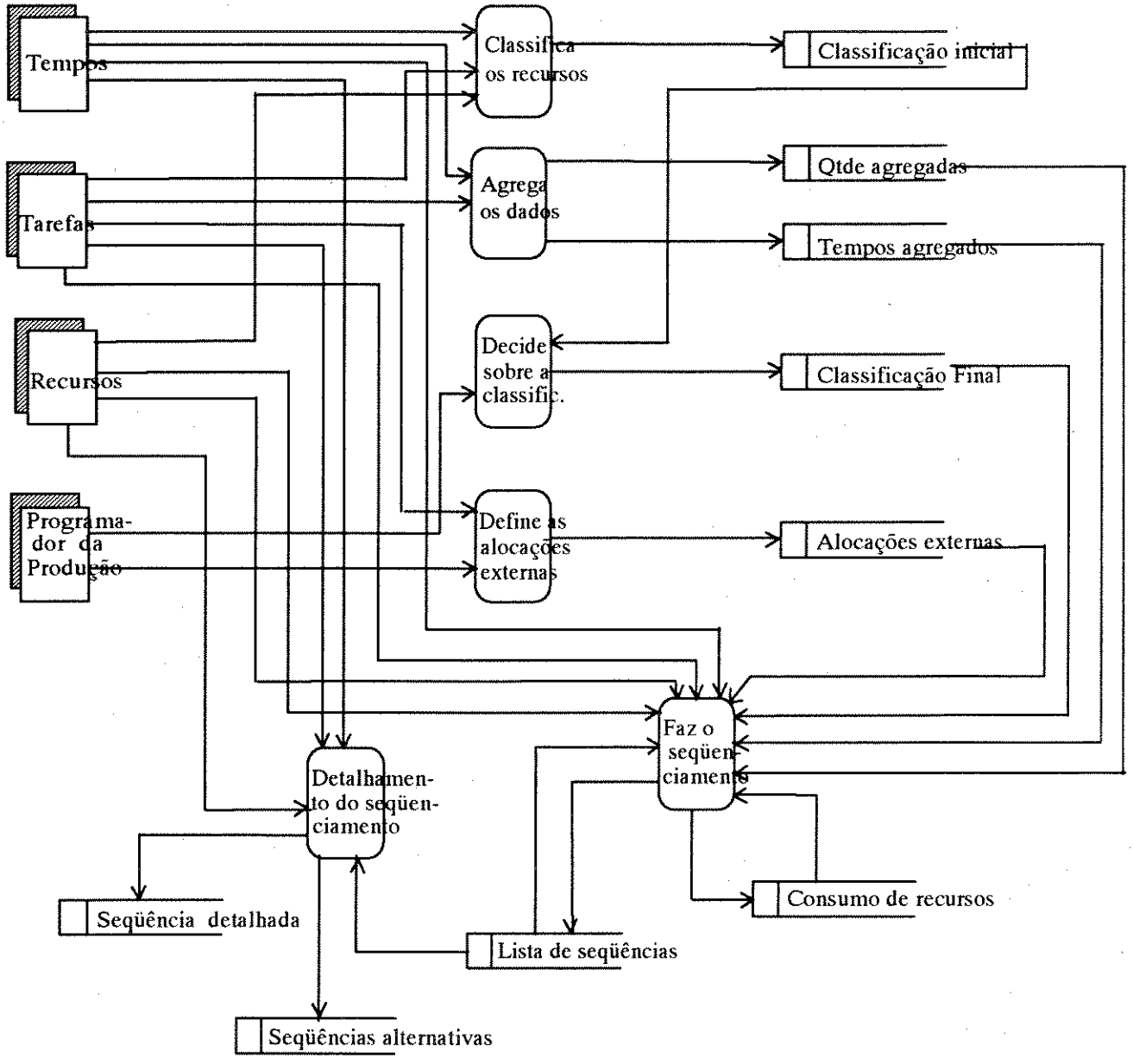


Figura A.2 : DFD da abordagem proposta

A.3.2 Descrição dos objetos

A descrição dos objetos ou classes contempla a definição de seus atributos e as operações sofridas e requeridas por outros objetos ou classes. Esta descrição mostra os principais aspectos definidos a nível do projeto. Alguns novos detalhes foram acrescidos posteriormente a nível de implementação, mas que por motivo de clareza na explicação não são aqui apresentados.

Tempos

Fazem parte de Tempos os seguintes atributos:

- tempo de preparação das operações das tarefas
- tempo de processamento das operações das tarefas
- tempo de transferência das operações das tarefas
- tempo total de execução das operações das tarefas

e as seguintes ações/interfaces :

- agrega tempos
- obtem tempo de preparação
- obtem tempo de processamento
- obtem tempo de transferência
- obtem tempo total

Tarefas

Fazem parte de tarefas os seguintes atributos :

- quantidade de tarefas a seqüenciar
- tarefas a seqüenciar
- tarefas alocadas externamente com as respectivas datas de início e fim
- perfil de consumo das tarefas

e as seguintes ações/interfaces :

- agrega perfis
- verifica factibilidade das tarefas alocadas externamente
- obtem tarefas
- obtem tarefas externas
- obtem perfil

Recursos

Fazem parte de Recursos os seguintes atributos :

- quantidade de recursos
- identificação dos recursos
- perfil de oferta dos recursos

e as seguintes ações/interfaces :

- obtem recursos
- obtem perfil

Classificação

Fazem parte de Classificação os seguintes atributos :

- classificação inicial
- classificação final

e as seguintes ações/interfaces

- faz classificação
- decide sobre a classificação
- obtem classificação inicial
- obtem classificação final

Seqüenciamento

Fazem parte de Seqüenciamento os seguintes atributos :

- lista de seqüências
- curvas de consumo (utilização) dos recursos
- número de iterações
- carta de Gantt

e as seguintes ações/interfaces

- executa algoritmo
- obtem lista
- obtem Gantt
- obtem curvas de consumo
- obtem estatísticas

Pós-seqüenciamento

Fazem parte de Pós-seqüenciamento os seguintes atributos :

- seqüência detalhada
- lista de seqüências alternativas

e as seguintes ações/interfaces

- detalha seqüência
- monta lista alternativa
- obtem seqüência detalhada
- obtem lista alternativa

Após a definição dos objetos a próxima etapa é a de detalhamento dos objetos, nesta etapa novos objetos podem ser criados. Um deles que é relevante do ponto de vista do projeto, é o que contempla os diferentes algoritmos de seqüenciamento (C1, C2 e C*). Estes algoritmos pertencem a subclasses da classe Algoritmo que contém uma definição básica dos algoritmos. Esta classe é considerada como sendo uma classe de base, a qual serve para a partir do mecanismo de herança definir as subclasses onde estão os algoritmos C1, C2, e C*.

A.3.3 Visibilidade dos objetos

A visibilidade de um objeto define o relacionamento existente entre dois objetos. Para representar a visibilidade são utilizados os chamados Diagrama de Objetos. Cada diagrama contém símbolos que denotam o seguinte :

- Objeto - o qual representa objetos distintos
- Visibilidade - o qual representa visibilidade entre objetos

e são representados nos diagramas através dos seguinte símbolos :

objeto

visibilidade

A visibilidade, definida no diagrama como um arco direto, denota um relacionamento entre dois objetos, expressados da seguinte forma : objeto A pode ver, mas não é visto, pelo objeto B. Para cada visibilidade, pode-se fornecer uma lista de itens os quais denotam um fluxo de controle ou dados que representam as operações que podem ser invocadas por um objeto sobre o outro, ou objetos que podem fluir entre dois objetos.

Tais diagramas são construídos incrementalmente, começando pela definição dos objetos e mais tarde colocando a visibilidade entre os mesmos. Novos objetos que forem

identificados, são também incluídos nestes diagramas. Estes diagramas são utilizados para visualizar os relacionamentos entre objetos e bastante úteis para analisar as implicações existentes quando são reestruturados.

A figura A.3 mostra o Diagrama de Objetos utilizado na fase de projeto do sistema aqui proposto.

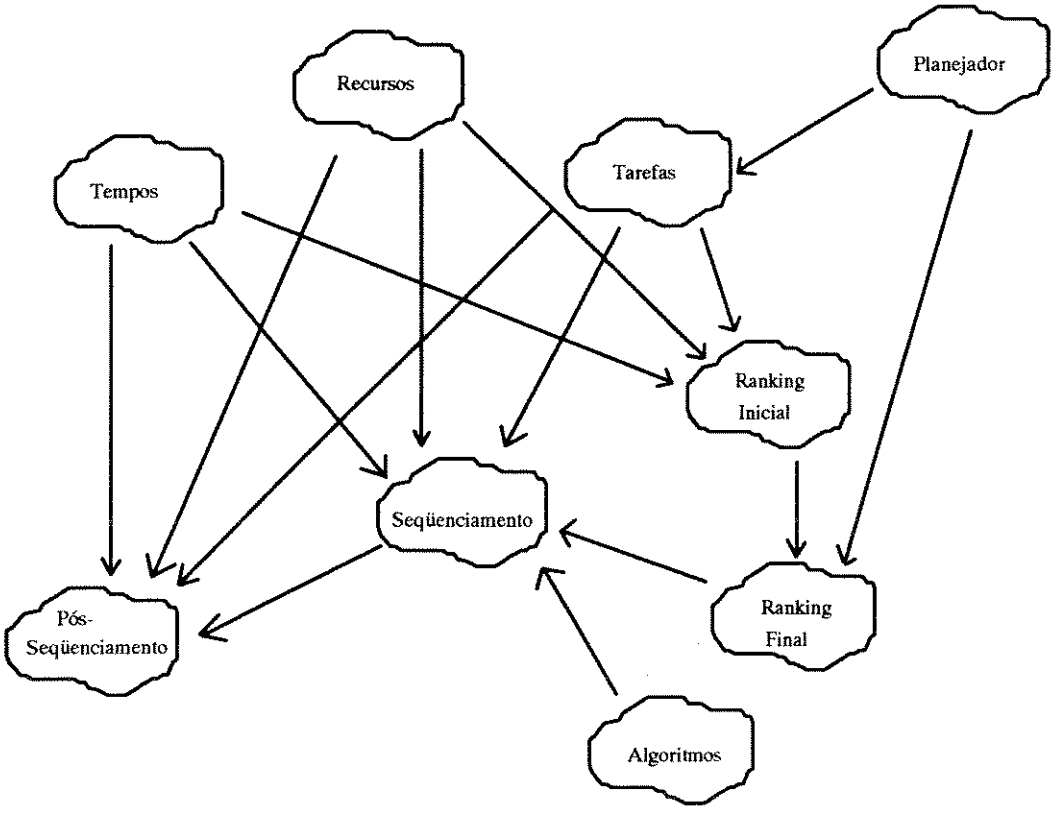


Figura A.3 : Diagrama de objetos

A.4 Discussão sobre a linguagem

A linguagem utilizada para a implementação da abordagem proposta foi a linguagem C++. Esta linguagem foi escolhida devido as suas características principais que passamos a discutir a seguir :

A linguagem C++ foi criada por Bjarne Stroustrup (Stroustrup (1986)) no início da década de 80. Stroustrup tinha dois objetivos principais : (1) C++ tinha que ser compatível com a linguagem C e (2) deveria estender a linguagem C com a construção de classes da linguagem Simula 67. Ambos os objetivos forma atingidos, e hoje em dia esta linguagem

vem sendo cada vez mais incrementada, principalmente por empresas do setor e Centros de Pesquisa que tem trabalhado seriamente para isto.

Tipos abstratos de dados ("Abstract Data Types" - ADT) são um dos elementos pertencentes à abordagem orientada ao objeto. Um ADT é uma extensão definida pelo usuário para os tipos definidos na linguagem. Ele consiste de um conjunto de valores e uma coleção de operações que podem agir sobre estes valores. C++ implementa ADTs através do mecanismo de herança entre as classes, onde um novo tipo pode ser derivado a partir de outro, através deste mecanismo. Desta forma, uma classe em C++ é a própria implementação de um ADT.

Classes em C++ são implementadas a partir do conceito de estruturas ("struct") em C. Em C++ estruturas podem ter função de "member". Estruturas podem ter também parte de sua descrição definida como privada. Estas extensões levam naturalmente ao conceito de classe que, em efeito, é uma estrutura com a visibilidade "default" de um "private". Este conceito de privacidade permite que parte da implementação fique escondida. Portanto, isto permite, que se defina também o que é visível externa e internamente por outras classes/objetos. Se isto for feito adequadamente a parte referente ao código privado pode ser alterada sem que isso implique em mudanças nas demais classes.

A linguagem C++, conforme a sua própria definição, contempla diversas características da linguagem C, que a tornam uma linguagem bastante flexível. Dentre estas características podemos citar:

- um programa pode usar vários arquivos, os quais podem ser compilados separadamente;
- não necessita de limites para vetores;
- usa bibliotecas standard para I/O;
- possui ainda diversas bibliotecas para funções chaves, tais como : manuseio de strings, arquivos, aritmética complexa, gráficos, etc;
- uma grande quantidade de operadores e muitos níveis de precedência.

C++ possui em sua definição os elementos essenciais para a programação orientada ao objeto. Esses elementos são os seguintes :

- recursos que permitam a implementação de ADTs;
- mecanismos de herança; e
- habilidade para processar objetos dinamicamente.

C++ é o casamento do baixo nível com o alto nível, onde o C permite trabalhar a nível de máquina, enquanto o C++ trabalha a nível do domínio. O programador pode escrever seus programas no nível apropriado para o problema, mas mantendo ainda contato, se necessário, com detalhes de implementação a nível de máquina. Isto faz com que se tenha uma flexibilidade muito grande para se programar eficientemente.

Em resumo, C++ suporta a programação orientada ao objeto. Sendo que a principal característica deste método é o encapsulamento de um conjunto apropriado de tipos de dados e suas respectivas operações (ADTs). Uma classe, com suas funções membros e operadores de "overloading", provê uma ferramenta apropriada de codificação. Classes também provêm mecanismos para esconder dados. Privilégios de acessos podem ser gerenciados e limitados para quaisquer grupos de funções que necessitem verdadeiramente acesso a detalhes da implementação. C++ foi influenciada pela linguagem Simula, uma linguagem específica para simulações, e isto vem de encontro ao paradigma Platônico: modelar ou simular o mundo concreto (Pohl (1991)).