UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

SEQÜENCIAMENTO DA PRODUÇÃO COM DIVISÃO EM CÉLULAS DE MANUFATURA E USO DE PROCESSAMENTO PARALELO

Marcio Mattos Borges de Oliveira

Orientador: Prof. Dr. José Francisco Ferreira Ribeiro

Tese apresentada à Escola de Engenharia de São Carlos , da Universidade de São Paulo, como parte dos requisitos para obtenção do título de Doutor em Ciências - Área: Engenharia Mecânica.

São Carlos, 1997

Ficha catalográfica preparada pela Seção de Tratamento
da Informação do Serviço de Biblioteca - EESC-USP

Oliveira, Marcio Mattos Borges de
O48s Seqüenciamento da produção com divisão em células de manufatura e uso de processamento paralelo / Marcio Mattos Borges de Oliveira. -- São Carlos, 1997.

Tese (Doutorado). -- Escola de Engenharia de São Carlos-Universidade de São Paulo, 1997.
Área: Engenharia Mecânica
Orientador: Prof. Dr. José Francisco Ferreira Ribeiro

1. Células de manufatura. 2. Processamento paralelo. 3. Produção. I. Título.

FOLHA DE APROVAÇÃO

Candidato: Engenheiro **MÁRCIO MATTOS BORGES DE OLIVEIRA**

Tese defendida e aprovada em 07-7-1997
pela Comissão Julgadora:

Prof. Doutor **JOSÉ FRANCISCO FERREIRA RIBEIRO (Orientador)**
(Instituto de Ciências Matemáticas de São Carlos - Universidade de São Paulo)

Prof. Doutor **CARLOS ALBERTO DOS SANTOS PASSOS**
(CTI - Campinas)

Prof. Doutor **EDUARDO MARQUES**
(Instituto de Ciências Matemática de São Carlos - Universidade de São Paulo)

Prof. Titular **JOÃO LIRANI**
(Escola de Engenharia de São Carlos - Universidade de São Paulo)

Prof. Doutor **EDUARDO VILA GONÇALVES FILHO**
(Escola de Engenharia de São Carlos - Universidade de São Paulo)

Prof. Assoc. **ARTHUR JOSÉ VIEIRA PORTO**
Vice-Coordenador da Área de Engenharia Mecânica
em exercício

Prof. Titular **JOÃO VITOR MOCCELLIN**
Vice-Presidente da Comissão de Pós-Graduação
em exercício

Senhor, tu tens sido o nosso refúgio, de geração em geração.
Antes que os montes nascessem e se formassem a terra e o mundo, de eternidade a eternidade, tu és Deus.
Tu reduzes o homem ao pó, e dizes: Tornai, filhos dos homens.
Pois mil anos, aos teus olhos são como o dia de ontem que se foi, e como a vigília da noite.
***Salmos*, 90**, 1-4.

O coração do homem pode fazer planos, mas a resposta certa dos lábios vem do Senhor.
Todos os caminhos do homem são puros aos seus olhos, mas o Senhor pesa o espírito.
Confia ao Senhor as tuas obras e os teus desígnios serão estabelecidos.
***Provérbios*, 16**, 1-3.

Agradecimentos

Agradeço ao Prof. Dr. José Francisco Ferreira Ribeiro pela inestimável orientação, dedicação e compreensão no desenvolvimento deste trabalho.

Agradeço à Profª. Drª. Cassilda Maria Ribeiro pela grande ajuda no entendimento do processamento paralelo.

Agradeço a todos os demais Professores que direta e indiretamente colaboraram para a execução desta tese repassando seus conhecimentos.

Agradeço o suporte financeiro dado pela CAPES na elaboração deste trabalho.

Agradeço a todos os meus alunos pela troca importante de informações ao longo destes anos.

Agradeço a todos os meus amigos, e não são poucos, que me incentivaram e ajudaram nesta jornada.

Agradeço imensamente à minha adorada esposa Sônia pela ajuda, estímulo e compreensão ao longo destes anos de trabalho.

Agradeço às minhas filhas Victória, Marília e Patrícia pela atenção que lhes foi roubada para a elaboração desta tese.

Aos meus queridos pais, José e Norma, que sempre me deram as condições e apoio necessários em todas as etapas de minha vida, dedico esta tese.

Índice

Lista de figuras

Lista de tabelas

Lista de siglas

Siglas	**Inglês**	**Português**
CAD	Computer Aided Design	Projeto Auxiliado por Computador
CAM	Computer Aided Manufacturing	Manufatura Auxiliada por Computador
CIM	Computer Integrated Manufacturing	Manufatura Integrada por Computador
CNC	Computer Numerical Control	Controle Numérico por Computador
FMS	Flexible Manufacturing System	Sistema Flexível de Manufatura
PC	Personal Computer	Computador Pessoal
AGV	Automated Guided Vehicle	Transportador automático de material
PERT	Program Evaluation and Review Technique	Técnica de revisão e programa de avaliação
CPM	Critical Path Method	Método do Caminho Crítico
VDSH	Variable-Depth Search Heuristic	Heurística de Busca com Profundidade Variável
LINDO	Linear, Interactive, Discrete Optimizer	Otimizador discreto, interativo e linear

Siglas	**Português**	**Inglês**
CN	Controle Numérico	Numerical Control
TG	Tecnologia de Grupo	Group Technology
CU	Controle de Usinagem	
CA	Controle Adaptativo	

Resumo

Um método para o seqüenciamento e a programação de operações baseado em células de manufatura, e uso de processamento paralelo é apresentado neste trabalho. A fábrica é organizada em células com o objetivo de decompor o problema global de seqüenciamento em subproblemas de dimensão reduzida. Esta decomposição permite uma simplificação das tarefas relacionadas ao controle e supervisão da fábrica. O processamento paralelo, por sua vez, possibilita maior rapidez aos cálculos e o uso do programa em tempo real.

Summary

A method to schedule and program operations based on manufacture cells and the use of parallel processing is presented in this study. The factory is organized in cells with the aim of decomposing the global problem of scheduling in subproblems of reduced dimension. This decomposition allows a simplification of tasks related to the control and supervision of the factory. Besides, parallel processing enables faster calculuses and the use of the program in real time.

Capítulo 1: Introdução

1.1 O Problema

Um sistema de produção pode ser definido como um procedimento de geração de bens ou serviços, através da transformação de recursos (Resende e Sacomano, 1991). Os sistemas de produção podem ser representados pela figura abaixo:

Entradas		**Processo de Conversão**		**Saídas**
Recursos	→	Transformação	→	Bens
Materiais	→	Máquinas	→	Conhecimento
Dados	→	Interpretação	→	Serviços
		Baseado em habilidade		
Custos Variáveis		**Custos Fixos**		**Receita**

Figura 1 - Esquema dos sistemas de produção

Os sistemas de produção combinam fatores como materiais, mão-de-obra, recursos (capital, energia, equipamentos, instalações, etc.), através de um processo organizado, com o objetivo de produzir algum bem ou serviço. Dentre os inúmeros sistemas de produção (fábricas, bancos, escritórios, supermercados, hospitais, etc.), esta tese dedica-se ao

estudo do problema de produção em fábricas, mais especificamente em fábricas onde a produção ocorre em quantidades pequenas de peças (lotes) e com uma variedade razoável de tipos diferentes de peças.

1.2 Hierarquia das Decisões

Quanto aos passos decisórios inerentes a um sistema de PCP-Planejamento e Controle da Produção, pode-se definir (Ackoff, 1975; Chiavenato, 1990), entre outros, os seguintes: longo prazo, médio prazo e curto prazo.

1.2.1 Longo Prazo

Em geral, os recursos fabris, equipamentos, mão-de-obra especializada e capital para estoque, não se tornam viáveis a curto prazo. Para tanto, as empresas devem elaborar planos de longo prazo com vistas ao dimensionamento das suas capacidades futuras. Neste nível elabora-se um planejamento com valores agregados de produção.

1.2.2 Médio Prazo

No médio prazo o planejamento agregado, gerado no nível anterior, sofre um processo de refinamento dando origem a um plano mestre de

produção, visando o dimensionamento da força de trabalho e os níveis de estoque.

1.2.3 Curto Prazo

Trata-se da determinação do plano global de produção para os próximos períodos e, portanto, de um guia para todas as operações de curto prazo. Ele decompõe o "Plano Mestre de Produção" em uma programação de produtos específicos a ser realizada em um período particular. Neste nível são realizados:

A) Planejamento de Materiais

É a atividade através da qual é feito o levantamento completo das necessidades de materiais para a execução do plano de produção.

B) Programação

O termo programação implica, de maneira geral, em aprazamento (Resende e Sacomano, 1991).Neste são realizadas as seguintes etapas:

1. Programação no Nível de Planejamento da Produção: elaboração do Plano de Produção.
2. Programação no Nível de Emissão de Ordens: acontece durante o processo de planejamento de materiais, envolvendo

reabastecimento, datas de término de fabricação e chegada de fornecimentos externos.

3. Programação no Nível de Liberação e Despacho: é uma fase de prazo muito curto e em um nível bastante detalhado. Determinam-se para cada ordem de fabricação, quando esta deve ter início e quanto é preciso trabalhar em cada operação planejada. Conhece-se o tempo de passagem de cada componente, que se compõe de tempo de processamento e montagem, tempos de movimentação e espera existentes entre cada operação.

O terceiro tipo de programação descrito acima é o objeto de estudo desta tese.

Na apresentação do trabalho realizado nesta tese, o termo programação é usado em referência à ordem de realização das operações e o termo seqüenciamento, quando além da ordem, define-se o momento no tempo (ou data) em que determinada operação será executada.

1.3 Conceitos

1.3.1 Categorias

A literatura apresenta três categorias de problemas de seqüenciamento, divididas segundo as restrições impostas às tarefas:

a) "*Flow-Shop*": todas as peças têm o mesmo roteiro de fabricação e passam nas mesmas máquinas. Tem-se um roteiro de fabricação único onde as p peças utilizam m máquinas na mesma ordem.

b) "*Job-Shop*": o roteiro de fabricação não é necessariamente o mesmo para todas as peças. Existe uma ordem para que as peças passem pelas máquinas. Este problema é matematicamente complexo e de difícil tratamento.

c) "*Open-Shop*": neste caso não temos ordem na fabricação. É um modelo menos restrito que o "job-shop" e o "flow-shop". A ordem de execução das tarefas componentes de um determinado trabalho é livre. Um trabalho é composto por m tarefas. Como exemplo pode-se citar a realização de exames médicos e reparos em oficinas mecânicas. A complexidade deste tipo de problema é considerável, pois não estão

disponíveis, por exemplo, os tempos das operações com precisão razoável.

A Tabela 1 fornece a cronologia dos estudos teóricos iniciais desta área. Hoje inúmeros estudos e métodos estão disponíveis na literatura, conforme o estado da arte fornecido em Ferreira Ribeiro et al. (1993), o que mostra o interesse pela área de pesquisa em questão, a evolução da pesquisa na área e as perspectivas de continuidade.

Tabela 1 - Cronologia dos resultados teóricos na área de *"job-shop"*

Ano	Autor(es)	Trabalho
1963	Muth e Thompson	Descrevem os primeiros resultados e propõem 3 exemplos para problemas de Job-Shop
1975; 1978	Mc Mahon e Florian; Carlier	resolvem os dois primeiros exemplos acima
Anos 80	Lageweg	resolve o terceiro exemplo
1986	Carlier e Pinson	Provam que a solução de Lageweg é ótima

1.3.2 Tecnologia de Grupo

A Tecnologia de Grupo (Mitrofanov, 1959) permite um melhor desempenho para as empresas que trabalham com produção de peças de forma unitária ou em pequenos lotes ("*job-shop*"). Para estas empresas, não se justifica a criação de uma linha de produção, pois esta caracteriza-

se pela baixa flexibilidade e é recomendada, em termos econômicos, unicamente para a produção em massa de um produto ou um conjunto pequeno de produtos diferentes.

Para a implantação da Tecnologia de Grupo é necessária a criação de uma base de dados com informações abrangentes do processo produtivo. A obtenção das informações relacionadas na Tabela 2, constitui uma das etapas mais difíceis a serem vencidas (Carlier, 1984).

Tabela 2 - Dados necessários

Máquinas	máquinas disponíveis por tipo
	relação de velocidades com referência a uma máquina tipo padrão
	período de trabalho
Peças	processo de fabricação: quais serão as máquinas usadas e em que ordem
	tempos de preparação ("*setup*") e de operação
	tamanhos de lotes

Todo o planejamento do sistema produtivo passa a ser em função das famílias de peças, representadas cada uma por uma "peça composta" fictícia reunindo todos os atributos das peças pertencentes à respectiva família.

A implantação da Tecnologia de Grupo ou Produção por Família de Peças é uma medida de forte impacto administrativo e gera a necessidade

de comprometimento de todos os departamentos envolvidos desde a concepção do produto até seu despacho para o cliente.

O conceito de Tecnologia de Grupo apresentado por Mitrofanov (1959) utiliza-se da definição de um "elemento composto", que seria a representação ideal de uma peça imaginária reunindo todas as características encontradas nas peças que compõe uma determinada família de peças. Os tempos de preparação das máquinas *("set-ups")* puderam ser reduzidos a partir desta nova concepção, o que proporcionou maior eficiência à fabricação.

Com estudos importantes nesta área, Burbidge (1975), deu continuidade a este trabalho, divulgando-o e realizando importantes avanços para implementação e desenvolvimento da técnica.

Em síntese, dada uma família de peças e o grupo de máquinas necessário para sua produção, forma-se o que se denominou de "Célula de Manufatura". Derivou desta abordagem o "Arranjo Físico Celular", que, por definição, refere-se ao grupo de máquinas convenientemente escolhidas, dimensionadas e arranjadas de tal forma que permitam a produção de todas as peças de uma família em seu interior.

Apesar da idéia básica do trabalho de Mitrofanov ter sido lançada há muito tempo, só mais recentemente é que as vantagens da Tecnologia de Grupo foram identificadas e reconhecidas. Esta demora se deve a três fatores básicos que, segundo Drucker (1997), somente a internacionalização total e a globalização da economia podem explicar satisfatoriamente:

1. Atualmente, a necessidade de utilização otimizada do equipamento disponível é prontamente aceita, dada à concorrência cada vez mais forte entre as empresas, decorrente da globalização. Busca-se uma contínua diminuição de custos com aumento de qualidade.
2. Para manter e conquistar novos mercados em escala mundial, as estratégias de Marketing apontam para a necessidade de diversificação maior das linhas de produtos, aumentando a variedade de peças a produzir em uma mesma fábrica.
3. Neste contexto, torna-se imperativo a diminuição dos custos de fabricação e o aumento da produtividade.

A Tecnologia de Grupo permite o gerenciamento da fábrica em células ou ilhas de fabricação (subsistemas mais simples) que tornam o

gerenciamento uma tarefa mais fácil face à gerência da fábrica como um todo. Identifica-se grupos de células que serão fabricados em subfábricas (constituídas pelas células de manufatura geradas) de maneira mais independente possível das demais.

Caso uma peça, por falta de máquina específica na célula de origem, necessite ir de sua célula a outra célula para realizar uma operação, tem-se um "movimento inter-célula". É bastante desejável que o número destes movimentos seja reduzido ao máximo, gerando células mais independentes de modo a facilitar o controle.

É importante observar que, de uma forma ou de outra, estes conceitos têm sido utilizados há mais de um século, como exercício da boa prática da engenharia, vindo somente a compor um conjunto coerente e sistemático de métodos aplicados à otimização da fabricação internacionalmente, há bem pouco tempo, conforme relata Sério (1990).

1.3.2.1 Outras Vantagens da Tecnologia de Grupo

A gestão de um setor onde a produção seja altamente variada e descontínua, pode ser alterada de maneira a aumentar a flexibilidade, a produtividade e o respeito aos prazos de fabricação pela adoção de um programa de implantação de células de manufatura. As tarefas também

podem ser simplificadas e a redução da burocracia e melhoria das relações humanas são outros fatores a serem considerados (Kusiak e Chow, 1988). Pode-se enumerar as seguintes vantagens advindas da introdução da Tecnologia de Grupo em uma fábrica:

1. O custo de fabricação de lotes pequenos e médios pode ser diminuído com a redução dos tempos de preparação das máquinas ("*set-up*") e da utilização mais efetiva de mão-de-obra especializada.
2. Redução dos custos de mão-de-obra com o operador controlando mais de uma máquina simultaneamente.
3. Redução dos custos de movimentação e transporte de peças, e a perda inerente de tempo, uma vez que as máquinas usadas na fabricação da família de peças se encontram próximas, na célula.
4. Como a célula se especializa em fabricar uma família de peças, tem-se a diminuição de custos com ferramental e demais dispositivos, visto que a padronização e a criação de ferramentas universais para a referida família torna-se viável.

Assim, com pequenos ajustes, pode-se usinar todas as peças de uma família.

Pode-se citar ainda: reduções dos níveis de estoque, de refugos, de custos de variedade de peças, do capital de giro, maior especialização de mão de obra, etc.

1.3.2.2 Aplicações da Tecnologia de Grupo

Inicialmente a usinagem de peças era feita em pequenos lotes, sendo as máquinas usadas na execução de apenas uma parte do trabalho ou processo global, como no caso das ferramentarias. A partir de 1913, Henry Ford e seu modelo T, introduziram a linha de montagem, e depois a linha de transferência. A partir de então, muitos conceitos foram desenvolvidos e estão em uso atualmente nas indústrias. Outros eventos importantes ocorreram a partir da introdução das máquinas de Controle Numérico (CN). De sua evolução surgiram os CN Computadorizados (CNC), Controle Adaptativo (CA), Centros de Usinagem (CU), Sistemas Flexíveis (FMS), Projeto de Fabricação com Auxílio de Computador (CAD/CAM) e Sistemas Integrados de Fabricação (ICAM) ou (CIM), Shtub (1989).

Dentro do desenvolvimento da indústria metal-mecânica, a Tecnologia de Grupo (TG) representou um grande avanço. Os fundamentos da TG, podem ser aplicados a todas as áreas de atividade de uma fábrica: produção, compras, administração, etc. A implantação de um sistema automatizado de manufatura é bastante facilitada com a organização prévia do arranjo físico ("*lay-out*") da fábrica em ilhas ou células de fabricação. Existem quatro fatores cruciais que suportam esta tese:

1. Volume de informação: sem a decomposição preconizada pela TG, o volume de informações normalmente grande, elevaria os custos com aquisição e gerenciamento de dados.
2. Sistema de transporte de material: a implantação de carregadores de materiais como os "*Automated Guided Vehicles*" (AGV's) ou robôs.
3. Requisitos Tecnológicos: as máquinas são agrupadas devido a exigências de ordem tecnológica.
4. Administração: a mão-de-obra humana, a despeito de toda a automação e do novo conceito em relação à manufatura clássica

será por longo tempo, parte fundamental deste novo sistema de manufatura.

1.4 Objetivo da Tese

O objetivo desta tese consiste em estudar o problema do sequenciamento de operações em uma fábrica do tipo *"job shop"*. Para tanto, dedicamos uma grande parte do trabalho ao estudo deste problema específico de produção e estabelecemos uma abordagem que envolve a decomposição lógica da fábrica em células de manufatura e a utilização da computação paralela como meio para estabelecer, em tempo real, uma seqüência e um calendário de fabricação para as operações a executar na fábrica.

1.5 Escopo da Tese

A decomposição da fábrica em células de manufatura e o uso da computação paralela, para contornar a complexidade computacional do problema de seqüenciamento e programação de operações em fábrica tipo "*job-shop*" permitiram um estudo pertinente e relevante para a área de pesquisa em questão. As experiências e resultados obtidos neste estudo levaram à publicação de resultados animadores e que ensejam a

continuidade do trabalho. Entre as publicações realizadas, podemos destacar: Oliveira et al. (1994a, b, c), Oliveira et al. (1996), Ferreira Ribeiro et al. (1996), Noronha et al. (1996).

1.6 Contribuição da Tese

Dada uma fábrica organizada em células de manufatura, a contribuição desta tese reside no uso da computação paralela para seqüenciar a produção e tratar os movimentos inter-células, definindo uma ordem e um calendário de fabricação, segundo um dado critério de otimização.

Em uma primeira instância, o estabelecimento de uma seqüência e de um calendário de fabricação é efetuado para cada uma das células de manufatura, atribuindo de modo biunívoco os cálculos a executar aos diferentes processadores disponíveis no computador paralelo. Em seguida, caso subsistam movimentos inter-células na partição lógica da fábrica em células de manufatura, efetua-se a transferência de informações entre os processadores e obtém-se o sequenciamento global da produção.

Esta maneira de estudar o problema do sequenciamento de operações é original e tem por objetivo a diminuição do tempo de cálculo dos algoritmos para poder-se operar nas fábricas com os programas computacionais correspondentes em tempo real.

Uma alternativa para a estratégia de resolução do problema de sequenciamento adotada nesta tese, mas sem o uso de processamento paralelo, seria a alocação de um microcomputador operando independentemente em cada célula de manufatura. Entretanto, esta alternativa exige, para um bom aproveitamento dos equipamentos, que a fábrica esteja fisicamente organizada em células e não contempla a análise dos movimentos inter-células e, assim, estes ficariam sem tratamento.

Capítulo 2: Revisão Bibliográfica

2.1 Seqüenciamento e Programação de Operações

O problema de seqüenciamento e programação de operações em uma fábrica com produção do tipo "*job-shop*" faz parte da classe de problemas combinatórios de difícil resolução: os problemas np-completos (Garey e Johnson, 1979). Dispõe-se de algoritmo de resolução polinomial unicamente no caso de problemas de dimensão reduzida, por exemplo, número de máquinas ≤ 2, para a definição de uma ordem de fabricação por meio do algoritmo de Johnson (1954). Esses problemas de pequeno porte podem, entretanto, ser resolvidos de maneira ótima por qualquer método: a busca exaustiva, por exemplo.

O modelo de Manne (1960) para o problema de se atribuir um intervalo do tempo disponível de uma máquina às operações a executar sobre as peças, apresenta um número de variáveis e restrições menor que os modelos propostos por Wagner (1959) e Bowman (1959). Sendo m o número de máquinas e p o número de peças, esta redução, no entanto, requer a determinação de $m*C_{p,2}$ variáveis bivalentes (Nemhauser e

Wolsey, 1988), o que impossibilita a resolução de um problema de porte médio em um tempo computacional aceitável.

Desta forma, o procedimento de busca exaustiva não pode ser utilizado para a resolução dos problemas de médio ou grande porte e o emprego de algoritmos aproximados, nesses casos, é uma técnica bastante difundida. Exemplos: busca de um ótimo local ou de uma melhor solução na vizinhança de uma solução inicial, decomposição (espacial, temporal, etc.) do problema a resolver em subproblemas de menor dimensão, realização de uma parada antes da obtenção da solução ótima por métodos exatos, etc.

Em Carlier e Chrétienne (1982) e Carlier (1984) são discutidos estudos sobre as diferentes versões existentes do problema de programação de operações: problema com uma máquina, problema com m máquinas, problemas com restrições unicamente relacionadas aos roteiros de fabricação, problemas com restrições disjuntivas, etc. Uma revisão dos métodos capazes de operar em tempo real é apresentada em Yamamoto e Nof (1985); sendo exemplos a citar neste caso os métodos propostos por Roubellat e Thomas (1988) e Ferreira Ribeiro e Pradin (1991).

A resolução do problema de seqüenciamento pelo método de separação e avaliação (branch-and-bound), tratando as disjunções em um grafo por meio de procedimentos heurísticos é proposta por Baker (1974) e Noronha e Ferreira Ribeiro (1994a, b, c). Nasr e Elsayed (1990) propõem dois algoritmos para a obtenção de uma solução aproximada por meio da decomposição do problema global em subproblemas mais fáceis de resolver. Gondran e Dostatni (1977) propõem uma regra heurística para a resolução dos conflitos de utilização das máquinas em um grafo disjuntivo. Dupont (1986) utiliza a técnica dos recozimentos simulados. No caso de problemas de grande porte, Portmann (1988) e Meguelati (1988) propõem uma parada antecipada após o processamento de um certo número de iterações por métodos exatos. Yamamoto (1977), Proth (1986), Didri e Proth (1987), Portmann (1988), Meguelati (1988), Staroswiecki et al. (1991), Ferreira Ribeiro e Pradin (1991), Ferreira Ribeiro e Ribeiro (1993), Ferreira Ribeiro et al. (1993) utilizam a decomposição espacial e/ou temporal do problema para seqüenciar subproblemas de porte reduzido. Nepomiastchy (1978) propõe um algoritmo de busca de ótimo local e a definição de uma programação admissível.

A busca arborescente por separação e avaliação (associada ou não a heurísticas) é uma das técnicas mais utilizadas para resolver o problema de programação de operações. Entre os métodos que a empregam, podemos citar: Ashour e Hiremath (1973), Gondran (1974), Carlier (1975, 1978, 1984), Yamamoto (1977), Yamamoto e Nof (1985), Hutchison e Chang (1990), Noronha e Ferreira Ribeiro (1994a, b, c). A programação dinâmica é proposta por Baker e Schrage (1978). A geração de um conjunto de planos de fabricação admissíveis ao fim de ajuda à decisão é proposta por Erschler et al. (1976), Erschler et al. (1979) e Roubellat e Thomas (1988).

Os métodos propostos na literatura podem ser divididos em duas grandes classes, segundo o critério de otimização adotado: Na primeira dessas classes agrupam-se os métodos que minimizam a duração total de fabricação (*"makespan"*) e, na segunda, os métodos que minimizam os atrasos de fabricação. Como exemplos de métodos da primeira classe, podem ser citados o PERT e o CPM (para problemas não-disjuntivos) e outros métodos, tais como Ashour e Hiremath (1973), Gondran (1974), Baker (1974), Kaufmann e Labordère (1974), Carlier (1975, 1978, 1984), Yamamoto (1977), Gondran e Dostatni (1977), Baker e Schrage (1978),

Yamamoto e Nof (1985), Dupont (1986), Hutchison e Chang (1990), Noronha e Ferreira Ribeiro (1994 a, b, c), etc. Na segunda classe, podem-se enumerar os métodos de Proth (1986), Portmann (1988), Meguelati (1988), bem como o algoritmo de Didri e Proth (1987) para estabelecer a seqüência ótima de produção em caso de máquina única.

Como exceção às duas classes citadas acima, podem-se citar o método de Nasr e Elsayed (1990) que minimiza o tempo de programação médio para execução das operações, o método de Nepomiastchy (1978) que minimiza um custo suplementar devido à violação das restrições, os métodos de Ferreira Ribeiro e Pradin (1991), Ferreira Ribeiro e Ribeiro (1993) e Ferreira Ribeiro et al. (1993), que permite a utilização dos dois critérios, e os métodos de Erschler et al. (1976), Erschler et al. (1979) e Roubellat e Thomas (1988), que não otimizam um critério, preferindo fornecer ao operador um conjunto de soluções compatíveis com as restrições a satisfazer, deixando-lhe a tarefa de selecionar a solução que lhe pareça a melhor. Um prazo de conclusão ou data de fim de fabricação mais tarde para os produtos (*"due date"*) é considerado nos métodos que procuram minimizar os atrasos de fabricação e também pelos métodos de Nepomiastchy (1978), Erschler et al. (1976), Erschler et al. (1979),

Roubellat e Thomas (1988), Ferreira Ribeiro e Pradin (1991), Ferreira Ribeiro e Ribeiro (1993), e Ferreira Ribeiro et al. (1993) nos quais se procura respeitar os prazos de entrega impostos aos pedidos. A atribuição de uma operação a uma máquina não é previamente fixada nos trabalhos de Erschler et al. (1976), Nepomiastchy (1978), Erschler et al. (1979), Roubellat e Thomas (1988), Meguelati (1988), Nasr e Elsayed (1990), Ferreira Ribeiro e Pradin (1991), Ferreira Ribeiro e Ribeiro (1993), Ferreira Ribeiro et al. (1993). Nestes métodos, conhece-se um conjunto de máquinas aptas a executar cada operação e utilizam-se as diferentes alternativas possíveis para estabelecer a programação.

Atualmente, existem no mercado pacotes integrados de *"software"* para Planejamento e Controle da Produção, tais como o MOOPI - Méthode d'Ordonnancement Otimisée pour la Production Industrielle (Método Otimizado de Seqüenciamento para a Produção Industrial). Isto demonstra o interesse que o mercado tem dado ao desenvolvimento de ferramentas para a otimização da produção. Especificamente sobre o MOOPI, o leitor pode encontrar maiores informações na Internet, no seguinte endereço: http://www.berclaim.com/frame.html.

Em relação a decomposição da fabrica em células de manufatura podemos citar contribuições de Rajagopalan e Batra, 1975; King, 1980; Vila Fo, 1982; Waghodekar e Sahu, 1984; Vanelli e Kumar, 1986; Kusiak e Chow, 1988; Shtub, 1989; Srinivasan et al., 1990; Gupta e Seifoddini, 1990; Venugopal e Narendran, 1992; Ferreira Ribeiro e Pradin, 1993; Ronconi e Armentano, 1993; Montevechi, 1995. Com relação à computação paralela podemos citar Amdahl, 1967; Schendel, 1984; Bouge, 1988; Quin, 1987; Hockney e Jesshope, 1988; Lavallee, 1990; Anshul e George, 1994; Sabot, 1995.

2.2 Carregamento de Máquinas e Projeto de Células de Manufatura

A determinação de uma ordem e de um calendário de fabricação, em nosso estudo, é precedida pelo processamento de duas fases:

- escolha das máquinas que vão executar as operações sobre as peças e atribuição das operações a executar sobre estas peças às máquinas escolhidas;
- organização do sistema de produção em células de manufatura, com o objetivo de decompor o problema global de

seqüenciamento e programação de operações em subproblemas de dimensão reduzida.

Nos testes realizados e que apresentamos no capítulo 6, o método e o programa computacional utilizado para resolver estes dois problemas é aquele proposto em Ferreira Ribeiro et. al (1996) para a primeira fase, e Ferreira Ribeiro e Pradin (1993) para a segunda.

O método de Ferreira Ribeiro et al. (1996) para a escolha e atribuição de máquinas é composto de duas etapas:

- Etapa de escolha das máquinas, onde é resolvido um problema da mochila por programação dinâmica (Minoux, 1983) ou através de um procedimento heurístico (Papadimitriou e Steiglitz, 1982) para gerar e classificar combinações de máquinas capazes de cobrir a totalidade da carga de trabalho existente sobre os tipos de máquinas;
- Etapa de carregamento dos lotes sobre as máquinas, onde é resolvido um modelo matemático em variáveis bivalentes através de busca por enumeração implícita (Salkin, 1975) ou através de um algoritmo guloso ("greedy"), para uma ou algumas

combinações de máquinas propostas pela etapa precedente, até que um carregamento factível seja encontrado.

Por sua vez, o método de Ferreira Ribeiro e Pradin (1993) para efetuar a partição da fábrica em células de manufatura é igualmente processado em duas etapas:

- Etapa de seleção das peças iniciais para formação das células de manufatura através de procedimentos específicos e de otimização da escolha realizada por meio da técnica das "nuvens dinâmicas" (Diday, 1971);
- Etapa de reatribuição das operações efetuada na implementação do módulo de seleção e carregamento de máquinas, de modo a eliminar ou diminuir ao máximo o número de movimentos inter-células da solução encontrada.

2.2.1 Seleção e Carregamento de Máquinas

Na área de seleção e carregamento de máquinas, numerosos outros estudos têm sido realizados nos últimos anos com o objetivo de se resolver de maneira eficiente versões específicas do problema de seleção e carregamento de máquinas. Amini e Racer (1994), por exemplo,

propõem o algoritmo aproximado VDSH (Variable-Depth Search Heuristic) para a resolução do problema da mochila multidimensional (onde cada máquina, a ser usada, é considerada como um problema da mochila tradicional) e fazem uma comparação de performance entre este e outros algoritmos aproximados descritos na literatura. A heurística VDSH é um método de busca local.

Ao contrário de Amini e Racer (1994), que resolve globalmente o problema da mochila multidimensional, Logendran et al. (1994) decompõem a resolução do problema de atribuição de peças a máquinas em 2 fases : Na 1ª fase determina-se, para cada tipo de máquina, o número de máquinas efetivas necessárias à fabricação das peças e, para cada peça, um único roteiro de fabricação entre os diferentes roteiros disponíveis. Na 2ª fase é realizada a atribuição. A busca proposta em Logendran et al. (1994) trata apenas a 1ª fase da metodologia. A 2ª fase não foi objeto de estudo.

Em Liao (1994), a seleção de um roteiro de fabricação para cada peça não é determinada por meio de uma heurística como é feito por Logendran et al. (1994). Liao (1994) faz a formulação matemática do problema correspondente com o auxílio de variáveis inteiras, levando em

conta o volume de produção e a capacidade das máquinas. A resolução do modelo formulado é realizada pelo pacote computacional LINDO, que não permite, evidentemente, o tratamento de problemas de médio ou grande porte. Sankaran e Kasilingam (1993) utilizam igualmente o pacote computacional LINDO associado a uma heurística para resolver o modelo de programação inteira proposto em seus estudos. Este modelo permite a determinação simultânea das máquinas que executarão as operações sobre as peças e das células de fabricação para um sistema flexível de manufatura.

Lashkari et al. (1987) utilizam o pacote computacional SAS/OR para resolver um modelo matemático em variáveis 0/1 com o objetivo de atribuir produtos a máquinas. Este modelo contempla o custo de depreciação das máquinas e a disponibilidade limitada das mesmas. A utilização do pacote SAS/OR foi possível pois as duas funções-objetivo propostas no modelo foram linearizadas.

Outras formulações em variáveis inteiras são estudadas em Wilhelm e Shin (1985), Shtub (1989) e Nasr e Elsayed (1990). Wilhelm e Shin (1985) propõem um modelo para efetuar a escolha entre diferentes máquinas, daquela que executará uma determinada operação. A resolução

do modelo é feita relaxando-se as restrições de integralidade e arredondando-se o resultado de modo a manter a factibilidade. Nasr e Elsayed (1990) propõem um modelo global para atribuição de produtos às máquinas e seqüenciamento de operações. O algoritmo de resolução proposto pelos autores faz uso da solução obtida para sucessivos problemas da designação (Gondran e Minoux, 1985).

Sule (1991) trabalha com uma tabela, onde é fornecida a porcentagem de capacidade disponível em cada tipo de máquina que é consumida pelas diferentes peças a fabricar. O número de máquinas necessárias, em cada tipo de máquina, para a fabricação das peças é igual à soma das porcentagens arredondada para o valor inteiro superior mais próximo da quantia obtida. Quando um número de máquinas calculado desta forma é capaz de fabricar globalmente as peças, mas não existe uma atribuição factível, avalia-se, por exemplo, o custo de se comprar uma nova máquina.

2.2.2 Projeto de Células de Manufatura

O crescente interesse pela aplicação da Tecnologia de Grupo nas indústrias pode ser avaliado, por exemplo, através dos estudos realizados

e descritos em Hyer e Wemmerlöw (1989) acerca de 32 companhias americanas que introduziram a Tecnologia de Grupo no chão de fábrica, em Arruda e Vila Fº (1995) sobre o estágio atual da implantação da Tecnologia de Grupo em 33 indústrias do Estado de São Paulo e no levantamento de Ferreira e Resende (1995) em outras 3 indústrias.

Os métodos de codificação e classificação agrupam as peças de acordo com características específicas, tais como : complexidade e forma geométrica, tipo do material, forma da matéria prima ou precisão do acabamento das peças. Usando um sistema de codificação, por exemplo: Opitz, Vuoso, Brisch, etc., as peças recebem um código numérico, alfabético ou alfanumérico. Cada dígito deste código representa uma característica da peça e as famílias são formadas realizando-se uma comparação entre os dígitos dos respectivos códigos. Um método que realiza o projeto de células de manufatura com base nesta técnica pode ser encontrada em Vila Fº (1982).

Na formulação matricial, uma matriz de incidência $[a_{ij}]$ constituída de elementos 0/1 é utilizada para representar o sistema de fabricação : a_{ij} = 1 (0) indica que a máquina j é usada (não usada) para processar a peça i. Os algoritmos de rearranjo matricial alteram de posição as linhas e

colunas de $[a_{ij}]$ com o objetivo de dar uma estrutura bloco-diagonal à mesma e definir as células de manufatura. Dentre os métodos de rearranjo matricial disponíveis na literatura, podemos citar : ROC-Rank Order Clustering (King, 1980), DCA-Direct Clustering Algorithm (Chan e Milner, 1981), BEA-Bond Energy Algorithm (McCormick et al., 1972), CBM-Cost Based Method (Askin e Subramanian, 1987) e CIA-Cluster Identification Algorithm (Kusiak e Chow, 1988).

O modelo matemático mais utilizado para auxiliar no projeto de células de manufatura é um modelo de programação inteira, denominado modelo da "p-mediana", quando dispõe-se de um único roteiro de fabricação para as peças e da "p-mediana generalizado", quando se dispõe de mais de um roteiro (Kusiak, 1987a). Neste modelo, procura-se minimizar a soma total das dissimilaridades entre as peças reunidas em uma mesma família e uma peça-semente, em torno do qual esta família está sendo formada. As restrições do problema asseguram que as peças serão atribuídas a uma única família, além disso, especificam o número desejado de células.

Entre os diferentes métodos que fazem uso dos coeficientes de similaridade, estão: MACE-Machine-Cell Formation in Group

Technology, de Waghodekar e Sahu (1984), ZODIAC- Zero-One Data: Ideal Seed Algorithm for Clustering, de Chandrasekharan e Rajagopalan (1987), e os algoritmos propostos por Rajagopalan e Batra (1985), Wei e Gary (1989), Shtub (1989), Xu e Wang (1989), Gunasingh e Lashkari (1989), Srinivasan et al. (1990) e Gupta e Seifoddini (1990).

Um estudo detalhado acerca dos coeficientes de similaridade é apresentado em Witte (1980). Outros modelos matemáticos de programação inteira descritos na literatura e que podem auxiliar na obtenção das células de manufatura são os seguintes: o modelo de programação quadrática (Kusiak e Chow, 1988), o modelo de programação fracionária (Lashkari et al., 1987) e o modelo de programação fracionária via transformação de Glover e Woolsey (1974), igualmente proposto por Lashkari et al. (1987).

Os métodos baseados em teoria dos grafos para efetuar o projeto de células de manufatura são, via de regra, algoritmos iterativos de coloração dos nós de um grafo G(N,A), onde N é o conjunto de nós, ou peças, e A é o conjunto de arestas, definido a partir de uma dissimilaridade crítica estabelecida entre as peças que devam ser atribuídas a diferentes células. Os algoritmos de Ferreira Ribeiro e

Ribeiro (1993), Ferreira Ribeiro e Alves (1994), Ferreira e Ferreira Ribeiro (1995), efetuam a coloração de G com o auxílio respectivo das seguintes técnicas : procedimento "greedy", técnica de "ligações-contrações" (Berge, 1973) e coloração de "árvores geradoras de peso máximo" (Szwarficter, 1983).

Lee et al. (1982) e Vanelli e Kumar (1986) efetuam a partição de um grafo representativo de um sistema de produção a partir de uma máquina definida como "máquina gargalo". Ronconi e Armentano (1993) utilizam o algoritmo de Gomory e Hu (1961) para particionar G e obter as células de manufatura. Barbosa e Ferreira Ribeiro (1994) propõem um procedimento heurístico iterativo de subdivisão cruzada de peças e máquinas, através da construção e avaliação de um grafo G, onde o critério de parada é a determinação de uma partição do sistema de manufatura com um número mínimo de movimentos inter-células.

A utilização de sistemas especialistas para se efetuar o projeto de células de manufatura é descrita, por exemplo, em Kusiak (1987a, b) e Kusiak (1988). Nestas implementações, considera-se roteiros de fabricação alternativos para as peças e a geração de cada solução parcial é realizada através de um algoritmo de partição selecionado

automaticamente pelo procedimento. Em Kusiak (1987b), procura-se determinar as células de máquinas e as famílias de peças, bem como selecionar um AGV (transportador automático de material), com correspondente custo mínimo, levando-se em conta que o tempo de processamento disponível em cada máquina, o limitante superior sobre a freqüência de viagens do AGV e um número máximo de máquinas em cada célula, não podem ser excedidos. Por outro lado, pode-se trabalhar com hipóteses de vários tipos, como a necessidade de se incluir determinadas máquinas nas mesmas células devido a necessidades tecnológicas.

A matriz representativa $a_{ij} = 0/1$ [peças x máquinas] de um sistema de produção é suscetível de partição por meio de métodos estatísticos em geral e, em particular, pela técnica de partição em torno de centros móveis, tais como o método das "nuvens dinâmicas" (Diday, 1971). Este método parte de peças-semente iniciais, uma para cada família de peças a ser formada, em torno das quais é induzida uma partição do sistema em células. Em seguida, escolhe-se outras peças-semente para induzir a formação de novas células. Estas peças são as que apresentam a menor soma de dissimilaridades com relação às outras peças atribuídas à família

da qual ela faz parte. O procedimento é repetido até que duas iterações sucessivas conduzam à mesma partição ou que o número de iterações atinja um número fixado previamente.

A maior parte dos trabalhos descritos na literatura para efetuar o projeto de células de manufatura faz uso de uma matriz de incidência binária $a_{ij} = 0/1$, ou seja : uma peça i utiliza uma máquina j e, neste caso $a_{ij} = 1$; caso contrário = 0. A técnica dos conjuntos nebulosos permite trabalhar-se com matrizes não-binárias e, por conseguinte, a introdução da incerteza usualmente encontrada na indústria quanto à máquina exata que vai realizar o processamento de uma determinada operação. Os trabalhos propostos por Montevechi (1995) e Montevechi e Gorgulho Jr. (1995) enfocam esta linha de pesquisa. Em Montevechi (1995), trabalha-se com uma matriz não binária, adaptando-se o conceito de dominante e dominado, e utilizando-se um valor de pertinência representativo da adequabilidade mínima de uma máquina executar ou não uma determinada operação presente no roteiro de fabricação de uma peça. A grande vantagem de se utilizar uma matriz não binária consiste em poder-se incluir máquinas alternativas para o processamento de uma mesma operação.

2.3 Modelos Matemáticos

A tendência atual coloca como imperativo a mudança nos sistemas de manufatura de forma a torná-los mais flexíveis e eficientes. A manufatura celular, ao contrário do sistema tradicional de organização de manufatura, pode aumentar consideravelmente os índices de produtividade e eficácia apresentados na fabricação de lotes pequenos e médios e sob encomenda.

Segundo Ferreira Ribeiro (1991), as estimativas apontam que 75% das peças produzidas nas indústrias metalúrgicas são executadas em lotes menores do que 50 peças. Este fato se deve à contínua diminuição dos ciclos de vida e diversificação das linhas de produtos, criando a necessidade de desenvolver novas técnicas de planejamento, conforme observa Pires e Agostinho (1988).

Este segmento industrial conhecido mundialmente como "*job-shop*", foi durante muito tempo subestimado ou tratado de maneira errônea com técnicas ligadas à manufatura de produção em massa (Pires e Agostinho, 1988).

A formação das células de manufatura pode ser feita de maneira otimizada. Dentre os vários critérios de otimização que podem ser usados para realizar esta tarefa, os mais importantes estão na Tabela 3:

Tabela 3 - Critérios de otimização

Critério de Otimização	minimizar	número de movimentos inter - células
		número de células
		duplicação de máquinas em diferentes células
		custo total de fabricação
		atraso de fabricação
	maximizar	utilização das máquinas
		operações de uma peça processadas dentro de uma única célula

Outras restrições podem ser adicionadas ao problema, tais como o número e a dimensão das células, bem como a capacidade das máquinas disponíveis.

2.3.1 Formulação para Projeto de Células de Manufatura

Organizar as células que irão compor uma fábrica não é tarefa simples e, em geral, exige a resolução de um problema matemático bastante complexo. Em Ferreira Ribeiro (1991) é proposta a seguinte formulação para a obtenção das famílias de peças:

PC01

Sendo:

x_{il} = atribuição da peça i à família constituída em torno da peça j.

d_{il} = dissimilaridade entre a peça i e a peça j.

$$\text{minimizar} \quad \sum_{i=1}^{\text{nº de peças}} \sum_{l=1}^{\text{nº de peças}} d_{il} * x_{il}$$

$$\text{sujeito a} \quad \sum_{l=1}^{\text{nº de peças}} x_{il} = 1 \quad (i = 1 \text{ a nº de peças}) \qquad (I)$$

$$\sum_{l=1}^{\text{nº de peças}} x_{ll} = \text{nº de células} \qquad (II)$$

$$dmin * x_{ll} \leq \sum_{i=1}^{\text{nº de peças}} x_{il} \leq dmax \quad (l = 1 \text{ a nº de peças}) \qquad (III)$$

$$x_{il} \leq x_{ll} \quad (i, l = 1 \text{ a nº de peças}) \qquad (IV)$$

$$x_{il} = 0/1 \quad (i, l = 1 \text{ a nº de peças}) \qquad (V)$$

A resolução do problema matemático em variáveis bivalentes zero-um PC01, dado acima, permite o particionamento das peças em famílias. Neste modelo, a variável bivalente x_{il} designa a atribuição da peça i à

família constituída em torno da peça j. A primeira restrição (I) assegura que uma peça pertencerá a uma única família de peças; a segunda (II) indica o número de famílias de peças que serão formadas; a terceira (III) dá as dimensões mínima e máxima (dmin e dmax) permitidas às famílias de peças; a quarta (IV) estabelece que a peça i só pertencerá à família de peças que contém a peça l se esta família existir; a última restrição (V) dá os valores possíveis para a variável x_{il}. A função-objetivo de PC01 minimiza a soma das dissimilaridades (d_{il}) entre as peças de uma mesma família com relação à peça escolhida como peça inicial desta família. Indiretamente, portanto, busca-se uma minimização do número de movimentos inter-células na fábrica.

As famílias de peças estando constituídas, as máquinas serão atribuídas às famílias de peças em que executam a totalidade ou a maior parte das operações.

2.3.2 Computação das dissimilaridades entre peças

As dissimilaridades entre duas peças i e l podem ser calculadas com base nas diferenças entre os seus roteiros de fabricação, como é feito em Kusiak (1987a, b, c), Wei e Gary (1989) e Ferreira Ribeiro e Pradin (1993). Dadas as peças i e l, definidas pelos vetores 0/1 :

$$P_i = [a_{i1}, \ldots, a_{im}, \ldots, a_{i\,nbm}]$$

$$P_l = [a_{l1}, \ldots, a_{lm}, \ldots, a_{l\,nbm}]$$

onde nbm = número total de máquinas disponíveis

e a_{im} = 1 se a peça i passa pela máquina m

0 caso contrário

Três situações podem diferenciar os roteiros de fabricação de duas peças i e l, com relação a uma máquina m:

peça i --> peça l -->	1 0	0 0	1 1
situação -->	I	II	III

No caso (I) a peça i passa pela máquina m e a peça l não; no caso (II) nenhuma das duas peças passa pela máquina m; em (III) as duas peças passam por m.

A dissimilaridade entre as peças i e l pode ser dada por:

$$d_{il} = \sum_{m=1}^{nbm} \upsilon[a_{im}, a_{lm}]$$

onde $\upsilon[a_{im}, a_{lm}] = a$ se $a_{im} \neq a_{lm}$

b se $a_{im} = a_{lm} = 0$

c se $a_{im} = a_{lm} = 1$

Kusiak (1987a, b) propõe a=0, b=0, c=1, Wei e Gary (1989) propõem a=0, b=1, c=nº peças-1 para o cálculo de similaridades entre peças. Ferreira Ribeiro e Pradin (1993) sugerem que a escolha seja feita pelo usuário, tendo em vista que a situação (I) - coeficiente "a" - traduz uma diferença entre os dois roteiros de fabricação que deve imperativamente ser penalizada no cálculo das dissimilaridades. A situação (II) - coeficiente "b" - traduz uma indiferença ($b \leq a$).

2.3.3 Seleção e Carregamento de Máquinas

Para efetuar a seleção e o carregamento das máquinas, Ferreira Ribeiro et al. (1996), resolvem o modelo matemático $PM01^j$, descrito abaixo, para cada tipo de máquina j.

PM01j

$$\text{minimizar} \quad \sum_{m=1}^{\text{nº de máquinas}} \Phi \left(\sum_{i=1}^{\text{nº de peças}} x_{im} \right) * c[j, m]$$

$$\text{sujeito a} \quad \sum_{i=1}^{\text{nº de peças}} carga[i, j] * x_{im} \leq cap[j,m] \quad (m=1 \text{ a } nm\_mq[j]) \quad \text{(I)}$$

$$\sum_{m=1}^{\text{nº de máquinas}} x_{im} = 1 \qquad (i = 1 \text{ a } nm\_lt) \qquad \text{(II)}$$

$$x_{im} = 0/1 \quad (i = 1 \text{ a nº de peças, } m = 1 \text{ a nº de máquinas}) \qquad \text{(III)}$$

onde: 1) $\Phi \left(\sum_{i=1}^{\text{nº de peças}} x_{im} \right) = 1$ se $\sum_{i=1}^{\text{nº de peças}} x_{im} > 0$

0 caso contrário

2) cap[j, m] = capacidade de produção da máquina de tipo j

3) carga[i,j] = carga de trabalho da peça i sobre o tipo de máquina j

4) c[j, m] = critério de otimização :

a) número de máquinas;

b) capacidade total das máquinas;

c) custo de utilização das máquinas

Capítulo 3: Seqüenciamento de Operações

3.1 Conceitos

Depois de expedida uma relação de ordens de produção para fabricação de peças ou lotes de peças, deve-se estabelecer um plano de produção de curto prazo e o processamento desta fabricação deve ser constantemente controlado. Com este procedimento pode-se assegurar uma melhor utilização dos tempos disponíveis, respeitar os prazos de entrega, melhorar a utilização dos equipamentos e mão de obra envolvidos, conseguir aumentos de produtividade, e garantir uma produção com eficiência e a preços competitivos.

Um seqüenciamento da fabricação consiste de uma lista de operações, atividades ou tarefas, onde cada operação será realizada numa certa ordem, com datas de início e término, determinadas por algum algoritmo. Define-se uma operação pela sua duração e ordem que ocupa no roteiro de fabricação do produto sobre a qual é executada.

Os dados necessários para a solução de um problema de seqüenciamento são:

- as operações e suas características,
- os recursos de materiais, mão-de-obra e equipamentos,
- as restrições,
- a função a ser otimizada.

Notação:

- I={1,2..., n} é o conjunto das operações,
- d[i] é a duração da operação i, sempre que esta duração não depender dos recursos que lhe são alocados,
- t[i] é a data mais-cedo para iniciar a execução da operação i,
- T[i] é a data mais-tarde para iniciar a execução da operação i,

Normalmente as tarefas têm relação entre si de anterioridade, caso contrário são chamadas independentes. A restrição de anterioridade, em sua forma mais geral, pode ser expressa por:

$$t[j] - t[i] \geq a(i,j)$$

Ela é válida sempre que $a(i,j) = d[i]$.

Existe o caso das operações poderem ser divididas em partes, sendo interrompidas e reiniciadas após a execução de outras tarefas.

Neste caso, a Programação Linear é o método mais indicado para se resolver o problema de programação (Hillier e Lieberman, 1988).

Podemos ter dois tipos de recursos: os renováveis e os consumíveis. Os recursos renováveis após terem sido usados em determinada tarefa estarão disponíveis integralmente para as outras. Podemos citar como recursos renováveis as máquinas, ferramentas, moldes, gabaritos, mão-de-obra, etc. Já os recursos consumíveis após sua alocação à determinada operação, não estão mais disponíveis para as demais operações. Temos como exemplos os recursos de capital, matéria prima, tempo, etc. Além disso, estes recursos podem estar disponíveis somente em alguns momentos. E pressupõe-se que a sua disponibilidade seja conhecida.

3.2 Restrições

Em algumas restrições de limitação de recursos é necessária a introdução de uma escolha sobre um conjunto de inequações. Isto se deve à necessidade de se resolver conflitos de utilização de recursos renováveis. Pode-se modelar o conjunto de conflitos pelo conjunto mínimo intermediário de tarefas que não podem ser realizadas

simultaneamente, tendo em consideração os recursos disponíveis. Estes são chamados conjuntos críticos de tarefas. Para resolver tal conflito, é suficiente ordenar duas tarefas quaisquer do conjunto crítico correspondente.

Um caso importante é aquele onde o cardinal do conjunto crítico é igual a dois, chamado de restrições disjuntivas (o par de operações [i,j] que constitui o conjunto crítico é chamado par de disjunção). Para resolver o conflito relativo a este conjunto crítico, i e j devem ser ordenadas num sentido ou no outro. Isto significa escolher entre duas desigualdades:

$$t[j] - t[i] \geq d[i] \text{ ou}$$

$$t[i] - t[j] \geq d[j]$$

Assim, quando somente interessa a resolução de conflitos relativos a conjuntos críticos de ordem dois, chamamos o problema de disjuntivo (utilização de máquinas-ferramentas, robôs, gabaritos, moldes, etc.).

Se existirem conjuntos críticos de ordem superior a dois, os problemas são chamados cumulativos. Neste caso será necessária a resolução de todos os conflitos. As restrições cumulativas são resultantes

da limitação das quantidades de recursos disponíveis, embora estas tarefas estejam prontas do ponto de vista de restrições potenciais.

Existem várias restrições para a elaboração de um plano de fabricação. Uma síntese destas restrições pode ser a seguinte:

- restrições potenciais: sendo i e j duas operações consecutivas sobre um mesmo produto, a operação j só pode ser iniciada após o término da operação i.

- restrições disjuntivas: a execução de um conjunto de operações não pode ser realizada simultaneamente se estas operações ocorrem numa mesma máquina.

- restrições cumulativas: os meios disponíveis para a execução de uma operação são limitados. O número de máquinas disponíveis em cada tipo pode restringir a execução de determinados planos.

A determinação de uma ordem e do calendário de fabricação serão definidos satisfazendo-se as restrições existentes e obedecendo um critério de otimização. Podemos citar os seguintes critérios mais usados:

- "*Makespan*" ou critério de minimizar a duração total de fabricação.
- "*Due Dates*" ou minimizar os atrasos de fabricação (maior atraso, soma dos atrasos, a soma ponderada dos atrasos, do número ponderado de operações em atraso), ou melhor dizendo a diferença entre as datas previstas de entregas e as planejadas para real fabricação.
- Minimização de um custo.

3.3 Formulações

Desde o final dos anos 50 vêm sendo desenvolvidos estudos para realizar a programação de operações com restrições disjuntivas por meio de modelos e técnicas de resolução propostos pela programação inteira.

Entre os modelos apresentados, Muth e Thompson (1963) citam: Wagner (1959), Bowman (1959) e Manne (1960), que utilizam o método dos planos de corte (Salkin, 1975) para resolução da formulação considerada. Balas (1969) aplica o algoritmo de resolução implícita (Salkin, 1975) para resolver o modelo proposto por Manne (1960). Este modelo apresenta um número de variáveis e restrições menor que o dos

outros dois modelos propostos anteriormente.

No início dos anos 70, a resolução dos problemas de programação com restrições disjuntivas sofreu um novo impulso com os trabalhos desenvolvidos na área da teoria dos grafos por Roy (1970) e retomados por Gondran (1974), Gondran e Dostatni (1977), Gondran e Minoux (1985), Carlier (1975, 1978, 1984), Carlier e Chrétienne (1982), Carlier e Pinson (1989).

3.3.1 Programação Inteira

As variáveis binárias (zero ou um) utilizadas nos modelos de Wagner (1959), Bowman (1959) e Manne (1960), citados por Muth e Thompson (1963) são:

- y_{ijk} = 1 se a peça i é a $j^{-ésima}$ peça processada na máquina k
  0 caso contrário (Wagner, 1959)

- x_{At} = 1 se a peça x é processada pela máquina A na $t^{-ésima}$ unidade de tempo
  0 caso contrário (Bowman, 1959)
- x_{ijk} = 1 se a peça j é processada antes da peça k na máquina i
  0 caso contrário (Manne, 1960)

Para a formulação de Manne (1960), os dados do problema são os seguintes:

a) m e p, para $\{i = 1, ..., m\}$ e $\{j = 1, ..., p\}$

b) d_{ij} = tempo de processamento da peça j na máquina i

c) roteiro de fabricação j(1), j(2), ..., j(m) para cada peça j

Seja t_{ij} o tempo inicial de execução da operação j na máquina i. Uma vez que a $(r+1)^{-ésima}$ operação sobre a peça j não pode ser executada antes que a $r^{-ésima}$ operação tenha sido completada, tem-se que:

$$t_{j(r+1),j} \geq t_{j(r),j} + d_{j(r),j} \text{ para } r = 1, ..., m-1 \text{ e para todo } j. \qquad (1)$$

A disjunção existente entre as operações j e k (j < k) sobre a máquina i é representada por meio de:

$$t_{ik} \geq t_{ij} + d_{ij} \text{ se } x_{ijk} = 1 \text{ ou } t_{ij} \geq t_{ik} + d_{ik} \text{ se } x_{ijk} = 0. \qquad (2)$$

Considerando M um número suficientemente grande ($M >> \max\{0, t_{ij}-t_{ik}+d_{ij}, t_{ik} - t_{ij} + d_{ik}\}$), transformam-se as restrições ou acima, em

restrições do tipo e:

$$t_{ij} - t_{ik} \leq d_{ij} + M\,(1 - x_{ijk}) \quad \text{e} \quad t_{ik} - t_{ij} \leq d_{ik} + M\,x_{ijk} \qquad (3)$$

Assim, o modelo para o problema é dado por:

minimizar $f = \Sigma\, t_{j(m),j}$

sujeito a : (1), (3), $t_{ij} \geq 0$ para todo i, j

e $x_{ijk} \in \{0, 1\}$ para todo i, j, k.

3.3.2 Teoria dos Grafos

Outra formulação importante para o problema de seqüenciamento e programação de operações é proposta pela Teoria dos Grafos.

Um grafo conjuntivo (ou seja, somente com restrições do tipo e) ou simplesmente um grafo G = (X, U) é um conjunto finito não vazio X e um conjunto U de pares de elementos de X (Berge, 1973), Gondran e Minoux (1986), Sakarovitch (1984).

Um grafo disjuntivo (ou seja, com restrições do tipo ou) é um grafo W = (G, D), no qual G = (X, U, d) é o grafo conjuntivo constituído do conjunto de nós X e do conjunto de arcos U cujos comprimentos d_{ij} são iguais às durações d_i das operações i que estão na origem dos arcos, e D é

o conjunto de restrições disjuntivas.

Uma restrição disjuntiva entre duas operações i e j é uma restrição do tipo ou e deve ser arbitrada de maneira a eliminar as disjunções, transformando W em um grafo conjuntivo: ou i é executada antes de j e insere-se, então, no grafo W um arco com origem em i e destino em j, de comprimento igual à duração da operação i, ou j é executada antes de i e insere-se, então, no grafo W um arco com origem em j e destino em i, de comprimento igual à duração da operação j.

Uma programação sobre o grafo disjuntivo W pode ser definida como um sistema de potenciais $T = \{t[i] \mid i \in X\}$ tal que: a) $\forall (i,j) \in U : t[i] - t[j] \geq d_i$, e b) $\forall (i,j) \in D : t[j] - t[i] \geq d_i$ ou $t[i] - [tj] \geq d_j$.

Uma arbitragem é um conjunto A de arcos disjuntivos tal que se $(i,j) \in A$, então $(j,i) \notin A$. Assim, o fato do arco (i,j) pertencer a A impõe que a operação i seja executada antes da operação j. À arbitragem A, associa-se o grafo conjuntivo $G(A) = (X, U \cup A)$, denotado por GA.

Uma arbitragem é dita completa se todas as restrições disjuntivas foram arbitradas. Uma arbitragem é dita compatível se o grafo conjuntivo associado à seleção A não contém circuito absorvente.

Uma solução para o problema é uma arbitragem completa e

compatível. A duração da solução obtida é o valor do caminho crítico do grafo conjuntivo associado. O grafo abaixo mostra um exemplo do problema do seqüenciamento. Este grafo foi gerado por uma versão do programa escrito em Pascal diretamente no AutoCAD R13. Nele, os arcos que ligam as operações em uma mesma máquina representam a ordem escolhida para entrada das peças nesta máquina. A criação destes arcos ocorre durante o processo de seqüenciamento

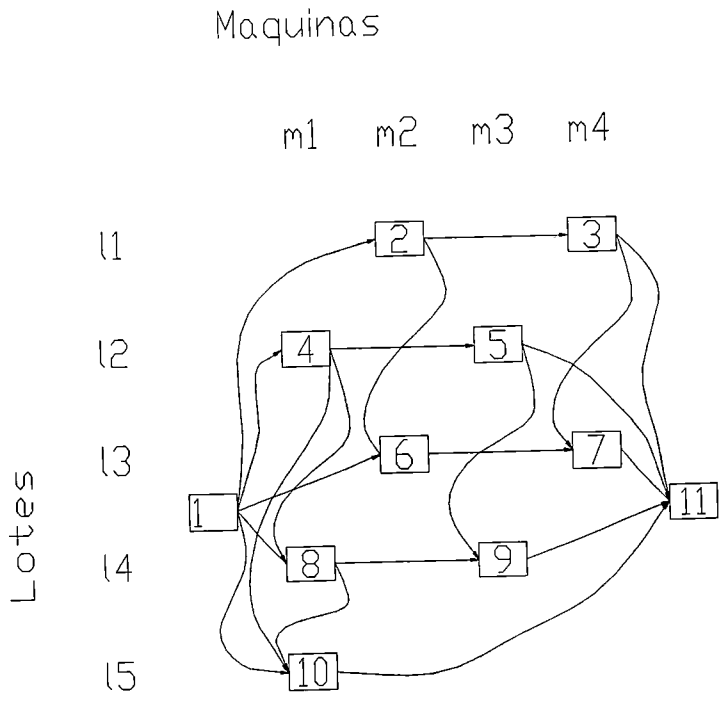

Figura 2 - Grafo de um seqüenciamento

3.4 Características e Aplicações do Seqüenciamento

Como já foi dito, esta tese tem seu estudo centrado no problema do "*job-shop*". Para melhor compreensão do problema, as características principais dos problemas de seqüenciamento estão enumeradas na Tabela 4.

Tabela 4 - Características dos problemas

Características	**Possibilidades (não exclusivas)**
Quantidade de meios	• ilimitado • limitado: renováveis e consumíveis
Organização dos meios	• uma máquina • m máquinas em paralelo • máquinas em série: "*flow-shop*" ou "job-shop" • máquinas em aberto: "*open-shop*"
Restrições de fabricação	• restrições potenciais (anterioridade) • restrições disjuntivas • restrições cumulativas
Tipos de fabricação	• unitário • pequena série • grande série (em massa) • contínua ou descontínua
Aspectos sobre a demanda	• estático (conhecida a priori) • dinâmico (chegada contínua em tempo real)
Critérios de otimização mais empregados	• minimização da duração total de fabricação • minimização dos atrasos de fabricação • minimização de um custo

As seguintes restrições para as p peças a serem fabricadas pelas m máquinas devem, por hipótese, ser observadas:

- uma máquina processa somente uma peça por vez;
- a passagem de uma peça numa máquina recebe o nome de operação;
- o roteiro de fabricação de uma peça é formado por operações;
- uma operação não pode ser interrompida ("*not preemptive*");
- saber a ordem de realização das operações nas máquinas é o objetivo da solução deste problema. Este objetivo pode ser alcançado usando-se critérios do tipo: duração total mínima, minimização dos atrasos, etc.

Capítulo 4: Algoritmos Paralelos e Computadores Paralelos

4.1 Introdução

A história do paralelismo no tocante a utilização de computadores paralelos ou de sistemas informátizados distribuídos para resolução de problemas concretos é muito recente.

Os primeiros computadores modernos surgiram durante a segunda guerra mundial. Eles se baseavam, por razões práticas, no princípio de Von Neuman, típico de um funcionamento seqüencial e na dependência causal das tarefas à executar. O computador executa uma instrução e em seguida o controle passa à instrução seguinte.

Disso resultou um desenvolvimento considerável de algoritmos numéricos seguindo o esquema seqüencial, sobretudo na área de ciências físicas (Ribeiro, 1991), assim como uma certa maneira de tratar os problemas.

Os primeiros artigos sobre paralelismo datam do fim dos anos 50. Com eles foram introduzidos os conceitos de paralelismo e concorrência, que levam em conta a independência causal de algumas tarefas e por conseguinte de tratamento simultâneo, não seqüencial e algumas vezes, de funcionamento não determinista.

Deve-se observar que até um passado bem recente o termo programação concorrente se aplicava somente aos sistemas operacionais e aos sistemas em tempo real.

4.2 Motivações para o Paralelismo

Entre os diferentes fatores que incentivaram o desenvolvimento e aplicação do paralelismo, pode-se destacar os seguintes : desempenho, modularidade, tolerância a falhas e os avanços recentes da microeletrônica (Marques, 1993).

Com relação ao desempenho, é evidente que quanto maior o número de processadores disponíveis, mais rápido, em princípio, tende a ser o processamento das informações. Deve-se observar que, os problemas advindos das comunicações entre processadores podem comprometer o desempenho esperado, dado que a aceleração

proporcionada pela elevação do número de processadores (*"speed-up"*) pode não acompanhar proporcionalmente o número de elementos incluídos.

A modularidade inerente ao paralelismo possibilita o agrupamento de módulos com o objetivo de se configurar sistemas de acordo com as necessidades específicas do usuário. Tal característica permite uma maior eficiência de operação.

As falhas no *"hardware"*, em muitos sistemas computacionais, não devem comprometer em definitivo o seu funcionamento. O paralelismo, dado o maior número de processadores, permite a continuidade da operação do *"software"* mesmo em situações de falhas no *"hardware"*.

Finalmente, os avanços da microeletrônica possibilitaram a construção de microprocessadores de alta performance, os quais podem ser reunidos e combinados para atingir a escala de processamento dos supercomputadores.

4.3 As Arquiteturas Paralelas

O desenvolvimento da tecnologia de circuitos integrados VLSI permitiu a evolução da arquitetura dos computadores e a construção de

novos computadores para responder às necessidades em potência de cálculo. Várias arquiteturas nasceram desta evolução: vetoriais (ou *"pipeline"*), sistólicas, *"dataflow"*, multiprocessadores à memória compartilhada e multiprocessadores à memória distribuída.

As arquiteturas vetoriais tem uma grande potência de tratamento devido à divisão das unidades funcionais em vários estágios. As operações vetoriais são decompostas em operações elementares. A cada etapa de cálculo várias operações elementares relativas a diversas componentes dos vetores podem ser efetuadas simultaneamente em estágios diferentes na unidade funcional utilizada.

Nas arquiteturas de multiprocessadores, o programa é dividido em tarefas independentes que são atribuídas à diferentes processadores. Estes processadores podem ser processadores vetoriais e seu modo de funcionamento pode ser síncrono, onde todos os processadores executam simultaneamente e de maneira seqüencial o mesmo processo, ou assíncrono e, neste caso, nenhuma condição será imposta sobre a execução e o encaminhamento das tarefas. A memória pode ser global (compartilhada) ou distribuída (local). No primeiro caso, as comunicações entre processadores são feitas por intermédio da memória global. No

segundo, as comunicações são feitas através de uma rede de interconexão que liga fisicamente os processadores.

As arquiteturas *"dataflow"* são arquiteturas onde a utilização dos processadores é guiada pelo fluxo de dados resultante do cálculo: as dependências causais entre os passos do cálculo guiam a utilização dos processadores. Cada processador busca um passo do cálculo que pode ser efetuado, executa-o e, em seguida, busca um outro passo a executar.

As arquiteturas sistólicas se situam entre as arquiteturas conexionistas e as arquiteturas *"dataflow"*. Elas possuem um grande número de processadores de baixo desempenho que são interconectados regularmente. A ativação dos processadores se faz por meio dos cálculos. As tarefas se deslocam ao longo da rede.

Há diferentes maneiras de se classificar as arquiteturas de computadores. A classificação apresentada a seguir foi proposta em Flynn (1966). Flynn construiu sua classificação sobre os dois conceitos clássicos de funcionamento de um programa de computador: o código a ser executado e os dados a serem tratados. Assim, nesta classificação, as máquinas são classificadas segundo o controle do fluxo de instrução.

Em um computador encontramos dois tipos de fluxos: I - fluxo de instrução *("instruction stream")* e D - fluxo de dados *("data stream")*. Estes fluxos podem ser do tipo S - único *("single")*, ou do tipo M - múltiplo *("multiple")*. As classes assim obtidas são as seguintes:

1) SISD ("Single Instruction Single Data")

Esta é a arquitetura dos computadores clássicos (seqüenciais). Ela executa uma só instrução a cada vez. A unidade de controle é única, mas pode dispor de várias unidades funcionais. Pode-se também usar o *"pipeline"*. A Figura 3 mostra um esquema desta arquitetura.

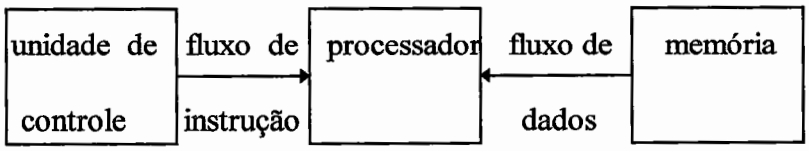

Figura 3 - Arquitetura SISD

2) SIMD ("Single Instruction Multiple Data")

A unidade de controle comum envia uma mesma instrução a todos os processadores.

Os processadores as executam simultaneamente e de maneira seqüencial sobre seus próprios dados.

O funcionamento é síncrono.

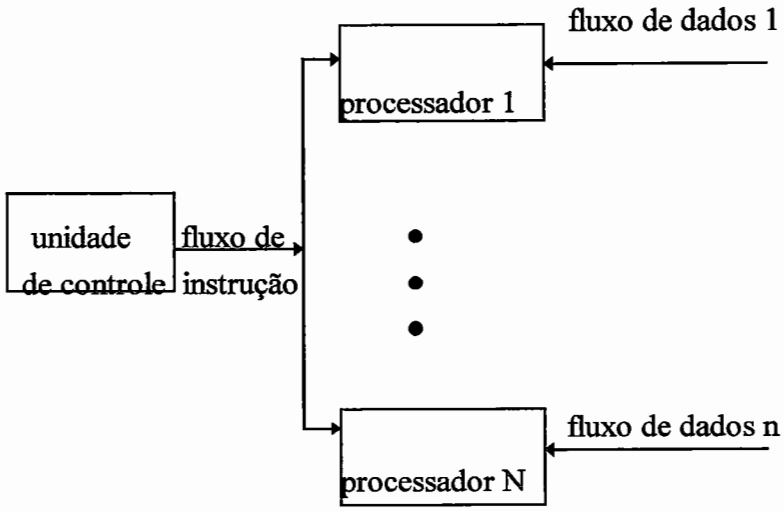

Figura 4 - Arquitetura SIMD

As arquiteturas SIMD são típicas de máquinas que dispõem de um grande número de processadores simples e de baixo desempenho (mais de 10000). Às vezes não se pode utilizar estes processadores isoladamente. Alguns só trabalham sobre um bit e são dotados de uma memória de milhares de bits (exemplo: *"connexion machine"*). Como resultado, os programas não podem ser armazenados na memória de cada processador.

O programa a ser executado é enviado aos processadores, instrução por instrução, pela unidade de controle global. Além disso, todos os processadores recebem a mesma instrução. Isto é sobretudo verdade para as instruções de transferência de informações, o que supõe uma interconexão regular entre os processadores do tipo grade. As transferências de informações são feitas de maneira ordenada, guiada pela geometria. Contudo, é difícil utilizar a potência das máquinas SIMD. É difícil conceber algoritmos em função das restrições geométricas de transferência de informação. Essas máquinas são dedicadas a certos tipos de aplicações particulares (tratamento de imagem, etc.).

Como exemplo de máquinas do tipo SIMD, podemos citar: ILLIAC IV, BSP, STARAN, MPP, DAP.

3) MIMD ("Multiple Instruction Multiple Data")

Existe uma unidade de controle e uma memória de tamanho pequeno ou grande (segundo a arquitetura adotada: a memória distribuída ou global) para cada processador. Os processadores executam tarefas diferentes com dados diferentes. O funcionamento é assíncrono. A Figura 5 fornece um esquema da arquitetura MIMD.

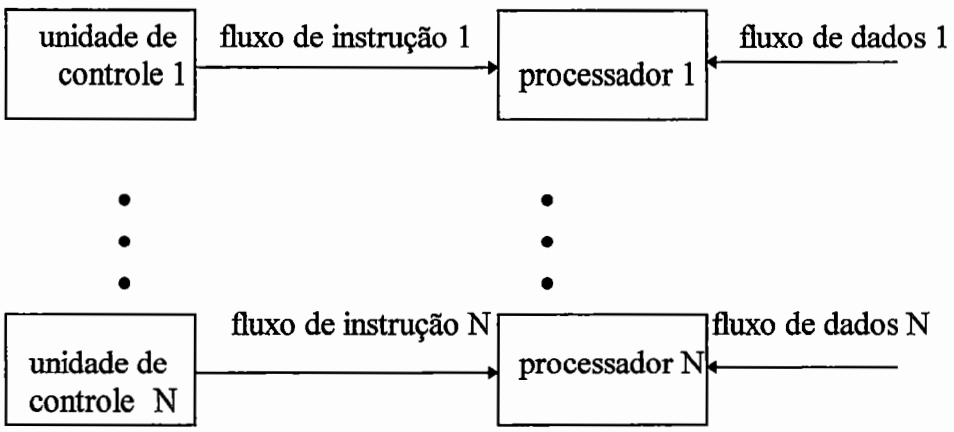

Figura 5 - Arquitetura MIMD

As arquiteturas MIMD são constituídas tipicamente por máquinas que dispõem de um pequeno número de processadores (inferior a 100) independentes e relativamente potentes (Exemplo: Hipercubo, IPSC da Intel). As transferências de informações são feitas de maneira desordenada, guiada pelos dados. Exemplos de máquinas MIMD: IBM 3090, Cray X-MP e Y-MP, hipercubos séries T de FPS e iPSC da Intel.

4.4 As Linguagens

O computador tradicional foi construído para a execução determinista de um programa seqüencial. Entretanto a busca por desempenhos maiores, rapidamente conduziu à introdução do paralelismo

(controle de entradas e saídas em paralelo, unidades de funções múltiplas, etc.). Contudo este paralelismo ficou muito tempo escondido do programador (pelo *"hardware"* ou *"software"*).

Os anos 80 e mais particularmente a segunda metade da década viu uma evolução particularmente importante em direção às arquiteturas paralelas ou distribuídas (supercomputadores vetoriais e paralelos, multiprocessadores, *"connexion machine"*, redes de estações, etc.): com as arquiteturas apareceu o conceito de programa não-determinista e por conseguinte, de gestão e controle desse não-determinismo.

Contudo, a evolução das máquinas paralelas foi mais rápida que a das linguagens paralelas. Além disso, o surgimento de linguagens paralelas se confronta com os hábitos de programação e sobretudo com a existência de códigos seqüenciais de tamanho considerável, que são usados quotidianamente e que não podem ser modificados na sua extensão.

A utilização de linguagens paralelas leva-nos a reconsiderar todos os aspectos metodológicos (gestão do paralelismo, não determinismo, comunicações, sincronização). O uso de linguagens paralelas implica em

uma maneira diferente de ver a especificação, a concepção e a validação dos programas.

Embora hoje existam máquinas paralelas, ainda não se sabe qual a melhor maneira de programar em paralelo, e algumas vezes não se busca descobrir, mas simplesmente obter um ganho de desempenho.

A problemática das linguagens de programação paralela consiste, então, nos seguintes aspectos (Bouge, 1988) :

- Qual semântica (significado, regra de evolução, critério de comparação) pode-se associar aos modelos fundamentais de paralelismo?
- Como exprimir o paralelismo, a comunicação e a sincronização nas linguagens de programação paralela?
- Como se pode especificar, provar, validar e avaliar estes algoritmos que são por natureza não-determinista?
- Qual a melhor forma de se utilizar os diversos tipos de arquitetura?

A seguir, são apresentados dois dos construtores mais conhecidos que foram propostos com objetivo de permitir aos programadores uma maneira explícita de indicar as partes de um programa que podem ser

executadas sobre os diferentes processadores de uma máquina à memória compartilhada.

4.4.1 Fork e Join: dois Construtores de Linguagem Paralela para Máquinas à Memória Compartilhada

Em 1965, ANDERSON propôs que se adicionasse as instruções FORK e JOIN à linguagem ALGOL afim de permitir o paralelismo (Williams, 1990). Em seguida, numerosas variantes de FORK e JOIN foram propostas.

FORK: inicia as tarefas paralelas.

JOIN: espera até que todas as tarefas estejam terminadas.

Exemplo:

```
FORK    tarefa 1, tarefa 2
JOIN    tarefa 1, tarefa 2
```

Uma instrução do tipo FORK pode ser considerada de duas maneiras diferentes:

1. FORK inicia um ou vários processos novos sobre um ou vários processadores e continua com o processo que executa o FORK sobre o mesmo processador. Neste caso, o processo que executa o

FORK é de uma certa maneira superior ao processo iniciado pelo FORK, uma vez que o primeiro processo gera o segundo.

2. FORK inicia vários processos novos sobre vários processadores, e o processador que executa o FORK já concluiu sua tarefa. Neste caso, os processos são iguais.

Uma instrução do tipo JOIN também pode ser considerada de duas maneiras:

1. JOIN termina um ou vários processos sobre um ou vários processadores, mas continua com o processo que executa o FORK. A instrução JOIN sincroniza os processos, pois ela consiste de uma espera até que todos os processos novos lançados estejam terminados.
2. JOIN termina vários processos e, então, um novo processo é iniciado. Neste caso, JOIN pode ser considerada como sendo a última instrução dos processos que estão terminando ou a primeira de um processo novo.

Existem várias maneiras de se ligar as instruções FORK e as instruções JOIN (Williams, 1990):

- Todo FORK deve ser acompanhado de um JOIN.

- Os processos que aparecem em um JOIN (Exemplo: JOIN tarefa 1, tarefa 2) devem figurar no mesmo FORK.
- Todos os processos existentes podem aparecer no mesmo JOIN.
- Os processos que aparecem em um FORK não precisam estar reunidos no mesmo JOIN. Exemplo:

```
processo A          processo B
.                   .
.                   .
FORK B              END
.
.
JOIN B
.
.
END
```

Neste exemplo, o processo A é executado sobre o primeiro processador até a instrução FORK. A instrução FORK autoriza o processo B a começar sobre um segundo processador enquanto o processo A continua a ser executado sobre o primeiro processador. Se o processo A atinge a instrução JOIN antes que o processo B esteja terminado, o processo A espera até que B termine e, em seguida, ele continua sobre o primeiro processador. Caso contrário, o processo A simplesmente continua sobre o primeiro processador.

A seguir, é apresentada a linguagem de programação OCCAM. Esta linguagem foi utilizada para implementar o algoritmo paralelo desenvolvido neste trabalho sobre um sistema cujos processadores elementares são os *"transputers"*.

4.4.2 OCCAM: Uma Linguagem de Programação para Arquiteturas do Tipo Transmissão de Mensagens

OCCAM (Hirsh, 1990) é uma linguagem paralela baseada sobre a formulação de processos seqüenciais comunicantes. Ela é baseada nas noções de concorrência e comunicação.

A linguagem OCCAM descreve a estrutura de um sistema constituído de processadores interconectados. Ela serve para programar cada processador e é também um formalismo de estudo, do mesmo modo que a álgebra de Boole é o formalismo de estudos para as portas lógicas (May e Taylor, 1984). Sua semântica formal permite ler um programa, seja como um conjunto de comandos, seja como um predicado numa extensão ao cálculo dos predicados.

A partir da semântica, um conjunto de regras foi definido para a transformação de programas. Estas regras podem ser utilizadas como base para a otimização de programas, autorizando assim o programador a

experimentar diversas implementações de um programa sabendo que estas implementações são equivalentes.

A) OS PROCESSOS

O processo é o elemento de base de uma aplicação OCCAM. Um processo é uma tarefa elementar e autônoma, com seus próprios dados e seu próprio código, que pode comunicar com outros processos que estão sendo executados ao mesmo tempo. Um programa é então um conjunto de processos.

De fato, todos os processos são construídos à partir de três outros processos elementares que são chamados de processos primitivos.

B) OS PROCESSOS PRIMITIVOS

Os processos primitivos da linguagem OCCAM são:

ATRIBUIÇÃO :

v := e

esta atribuição fornece à variável v o valor de expressão e.

ENTRADA :

c ? v

esta entrada fornece à variável v o valor que entrou no canal c.

SAÍDA :

c ! e

esta instrução faz com que o valor da expressão e saia no canal c.

Os dois últimos processos primitivos são utilizados para a comunicação entre processos.

As comunicações entre processos concorrentes são efetuadas unicamente com a ajuda de canais. Cada canal fornece uma conexão unidirecional entre dois processos concorrentes. Um processo envia os dados num canal (saída), e o outro processo recebe os dados do canal (entrada).

As comunicações são sincronizadas. Quando um processo comportando uma entrada e um processo comportando uma saída estão prontos para se comunicar sobre o mesmo canal, o valor a sair é copiado do processo de saída no processo de entrada. Em seguida, cada procedimento continua. Em função de sua estrutura, um processo pode efetuar várias comunicações simultaneamente.

OCCAM pode ser utilizada para programar uma rede de computadores. Cada computador com uma memória local executa um

processo com as variáveis locais. Cada conexão entre dois computadores é implementada por um canal entre estes dois processadores.

C) CONSTRUTORES

Vários processos, primitivos ou não, são combinados para formar os processos maiores, com a ajuda de construtores. Estes construtores são compostos de uma palavra reservada que estipula como os processos a serem reunidos devem ser executados (paralelo, seqüencial ou alternativo). Os diversos construtores são definidos abaixo:

O construtor SEQ :
Exemplo :

```
VAR x:
SEQ
   input ? x
   output ! x
```

O construtor seqüencial SEQ provoca a execução seqüencial de seus componentes. Um processo começando por um SEQ termina quando seu último componente termina.

O construtor PAR :
Exemplo :

```
CHAN OF INT comm:
PAR
  INT x:
  SEQ
    input ? x
    comm ! x
  INT x:
  SEQ
    comm ? x
    output ! x
```

O construtor paralelo PAR provoca a execução concorrente de seus componentes. Ele pára quando todos seus componentes tiverem terminado.

O construtor ALT :
Exemplo:

```
CHAN OF INT chan1, chan2 :
INT x:
ALT
  chan1 ? x
    processo1
  chan2 ? x
    processo2
```

O construtor do tipo alternativo, ALT, provoca a execução de somente um dos seus componentes. Ele termina quando a componente escolhida chega ao fim. Cada componente de um ALT começa por um processo de entrada, também chamado de dispositivo protetor (*"guard"*).

A componente que é escolhida para ser executada corresponde ao primeiro processo que está pronto para ser executado. Se vários dispositivos protetores estão prontos, uma escolha arbitrária é efetuada.

Em OCCAM temos também os construtores clássicos IF e WHILE.

Um programa paralelo em OCCAM é então construído utilizando-se variáveis e canais em processos primitivos, que em seguida são combinados em construções mais complexas.

4.5 Problemática do Cálculo Paralelo

A chegada das máquinas ditas paralelas impôs uma mudança na área de métodos numéricos, para poder tratar de maneira concorrente as partes independentes de um mesmo programa. A solução numérica de um problema, no contexto sequencial, é composto de três etapas:

1. escolha do algoritmo
2. implementação do algoritmo em uma máquina
3. estudo de desempenho

No contexto do cálculo paralelo, a solução numérica deste mesmo problema é essencialmente composta de seis etapas:

1. escolha do algoritmo
2. divisão em tarefas
3. atribuição das tarefas aos processadores
4. sincronização das tarefas
5. implementação do algoritmo paralelo em uma máquina
6. estudo de desempenho

Em consequência, a problemática do paralelismo é constituída, em parte, por problemas do tipo matemática aplicada e, em parte, por problemas de natureza informática.

4.5.1 Escolha do Algoritmo

O algoritmo escolhido deve colocar em evidência e tirar proveito com o paralelismo natural do problema a ser resolvido. Para escolher um algoritmo, tem-se as seguintes alternativas:

1. Conceber um algoritmo paralelo de alto desempenho adaptado a uma dada arquitetura paralela. Em certos casos, pode-se mesmo propor a arquitetura paralela que melhor se adapte à implementação do algoritmo paralelo.

2. Selecionar um algoritmo sequencial em função do seu desempenho, complexidade algoritmica e de sua aptidão a ser paralelizado.

4.5.2 Divisão em Tarefas

Deve-se procurar as tarefas importantes que podem ser executadas por processadores independentes, de modo a minimizar as ligações de casualidade entre tarefas destinadas a ser implementadas em processadores diferentes. Para isso, é preciso fazer um estudo da complexidade algoritmica e da complexidade em comunicações. Na realidade, a etapa de divisão das tarefas não é desconectada da escolha do algoritmo.

4.5.3 Atribuição das Tarefas aos Processadores

Depois da partição das tarefas, deve-se atribuir cada tarefa a um ou vários processadores para sua execução. Cada processador recebe, então, um subconjunto de tarefas.

Existem duas maneiras de se atribuir as tarefas:

1) modo estático;

2) modo dinâmico.

No modo estático, cada processador é atribuído de maneira definitiva a um conjunto de tarefas. Isto tem a vantagem de minimizar as transferências de dados no caso da utilização de arquiteturas paralelas, onde cada processador possui uma memória local (de fato, neste caso, as informações relativas às tarefas podem ser conservadas nas memórias locais). O inconveniente deste modo reside na intolerância a falhas. Se uma falha ocorre, o algoritmo não pode mais ser executado.

No modo dinâmico, cada processador é atribuído sucessivamente a um conjunto de tarefas diferentes. Isto tem o inconveniente de multiplicar as transferências de informações. A vantagem reside na tolerância a falhas. Com efeito, enquanto existir um processador que funcione sem falhas, o algoritmo pode continuar a sua execução.

A atribuição das tarefas deve ser efetuada de modo a equilibrar as cargas de cálculo nos processadores.

4.5.4 Sincronização das Tarefas

Trata-se de especificar uma ordem de execução das tarefas que garanta convergência, a rapidez de cálculo, etc. Para tanto, pode-se escolher diversos esquemas de evolução do conjunto de tarefas: totalmente sincronizadas, parcialmente sincronizadas, assíncronas. Em decorrência, é preciso estudar as propriedades de convergência destes esquemas, as taxas de convergência, etc. : Miellou (1975), Miellou (1986), Bertsekas e Tsitsiklis (1989), El Baz (1990), Tseng et al. (1990), Ribeiro e El Baz (1992).

Os parâmetros importantes a levar-se em conta quando da escolha do tipo de sincronização são os seguintes: a natureza particular do problema, o tipo de arquitetura informática utilizada e o desempenho da máquina.

4.5.5 Implementação do Algoritmo Paralelo em uma Máquina

A implementação do algoritmo concerne todos os aspectos informáticos ligados à utilização de um algoritmo paralelo sobre uma máquina paralela. Por causa disto, deve-se notadamente, observar a

codificação do algoritmo paralelo, a programação das comunicações entre processadores, o funcionamento das sincronizações, a reconfiguração da rede, o teste de parada do algoritmo e a proteção dos dados.

4.5.6 Estudo de Desempenho

Há vários fatores que podem ser significativos para o estudo do desempenho de uma implementação paralela. Entre estes fatores, pode-se citar:

- A influência do sistema:
  - tipo de arquitetura
  - número de processadores, topologia da rede de interconexão
  - características dos processadores, da rede, das memórias
- A influência dos diferentes parâmetros do algoritmo
  - equilíbrio das tarefas
  - dimensão das tarefas
- A comparação com outros algoritmos paralelos

Uma outra maneira de analisar o desempenho dos algoritmos paralelos é o "fator de aceleração", notado S_p (*"speed-up"*, em inglês).

O *"speed-up"* é uma medida de desempenho estabelecida através da comparação entre o algoritmo paralelo e o sequencial. Ele permite medir o ganho de tempo devido à utilização de p processadores (Schendel, 1984) :

$$S_p = T_1 / T_p$$

onde:

p - corresponde ao número de processadores

T_1 - tempo de cálculo sequencial

T_p - tempo de cálculo com p processadores

Denomina-se "eficiência" a medida da taxa média de utilização de p processadores. Ela é dada por:

$$E_p = S_p / p$$

Capítulo 5: Algoritmos para Seqüenciamento

5.1 Seqüência e Calendário de Fabricação

Dispõe-se de um parque de máquinas de diferentes tipos j, j = 1 a m, existindo para cada tipo um certo número de máquinas disponíveis, e para cada máquina, a relação de velocidade de desempenho com relação à máquina de referência do tipo. Trata-se de planejar a fabricação de lotes de produtos i, i = 1 a n. A cada lote de produtos está associado um roteiro de fabricação (lista ordenada de operações - ou tipos de máquinas - a executar sobre os produtos do lote) e para cada operação do roteiro, uma duração sobre a máquina de referência do tipo. Cada lote de produtos é igualmente caracterizado por uma data de início de fabricação mais cedo; por exemplo, a data de disponibilidade da matéria prima na fábrica, e por uma data limite de fabricação mais tarde; a partir da qual qualquer atraso comprometerá a data de entrega do lote fabricado.

Após o processamento das etapas precedentes (I) atribuição e (II) organização da fábrica em células de manufatura, utilizando, neste caso, o método proposto por Ferreira Ribeiro e Pradin (1993), os lotes de produtos estão distribuídos às diferentes células e possuem um roteiro de fabricação com lista ordenada das máquinas utilizadas e as durações de cada operação efetiva sobre as máquinas correspondentes. Cada lote de produtos estando caracterizado por uma data de início de fabricação mais cedo (inicio[i]) e por uma data de fabricação mais tarde (fim[i]), resolve-se o seguinte problema de seqüenciamento: Para cada operação k executada em cada célula de manufatura, calcular a data de início de fabricação mais cedo (mais_cedo[k]) e uma data de fabricação mais tarde (mais_tarde[k]).

5.2 Seqüenciamento das Operações nas Células

O procedimento de seqüenciamento proposto é baseado na construção e análise de um grafo (Gondran e Minoux, 1986) : Para cada célula p associa-se um grafo $G_p = (N_p, A_p, l_p)$, onde N_p é o conjunto dos nós do grafo, A_p é o conjunto dos arcos e l_p o conjunto dos comprimentos associados aos arcos. Os nós N_p representam as

operações a realizar na célula p, mais 2 operações fictícias chamadas opα [p] e opω[p]. Um arco liga duas operações efetivas e sucessivas k_1 e k_2, e tem por comprimento a duração da operação k_1. O nó opα[p] é ligado ao nó associado à primeira operação sobre cada produto i por um arco de comprimento início[i]. Os nós associados à última operação sobre cada produto i são ligados ao nó opω[p] por arcos de comprimento igual à duração destas operações.

As datas de início de execução mais cedo e mais tarde para as operações são calculadas segundo o seguinte procedimento: A data mais_cedo[k] é o comprimento do caminho máximo entre opα[p] e o nó k. A data mais_tarde[k] é a diferença entre o mais_tarde[opω[p] e o comprimento do caminho mínimo entre o nó k e o nó opω[p]. A duração mínima da fabricação do conjunto de produtos será então o comprimento do caminho máximo entre os nós opα[p] e opω[p], o comprimento de um caminho entre 2 nós sendo definido como a soma do comprimento dos arcos que compõem este caminho. A data mais_tarde[opω[p]] pode ser fixada de 2 diferentes maneiras:

1. **"Clássica":** após calcular as datas mais_cedo[k], faz-se mais_tarde[opω[p]] = mais_cedo[opω[p]]. Utiliza-se esta forma quando se procura minimizar a duração total do seqüenciamento, sem considerar as datas de entrega dos produtos acabados Gondran e Minoux (1986).

2. **"Alternativa":** mais_tarde[opω[p]] tem um valor diferente para cada lote de produtos: mais_tarde[opω[p]] = fim[i]. Esta forma considera as datas de fim de fabricação mais_tarde, o que permite evidenciar atrasos existentes nas execuções das operações (Ferreira Ribeiro, 1991).

As operações realizadas sobre uma mesma máquina constitui uma "clique de disjunção" e não podem ser executadas simultaneamente. Assim, arcos devem ser introduzidos entre os nós pertencentes a uma mesma clique, de maneira a estabelecer uma ordem de passagem das operações sobre a máquina e fixar o calendário de fabricação. A resolução das r primeiras disjunções (r fixado pelo usuário) é feita com a

ajuda da Regra I fornecida abaixo. Em seguida utiliza-se a Regra II, mais rápida que a Regra I:

- **Regra I:** Implementa-se um método arborescente, escolhendo-se como dupla de operações a separar, aquela cuja penalidade r[x,y] apresente valor máximo. Em caso de empate, adota-se aquela correspondente ao p[x,y] máximo. Os coeficientes r[x,y] e p[x,y] são dados por: p[x,y] = min {w[x,y], w[y,x]} e r[x,y] = | w[x,y], w[y,x] |, onde w[x,y] = max{0, mais_cedo[x]+duração[x]-mais_tarde[y]} e w[y,x] = max{0, mais_cedo[y]+duração[y]-mais_tarde[x]}. [Gondran (1974)]

- **Regra II:** Entre as operações k pertencentes a uma mesma clique de disjunção, executa-se em prioridade, aquela cuja soma (mais_cedo[k] + mais_tarde[k] + duração[k]) apresentar o valor mínimo. [Gondran e Dostatni (1977)]

Para resolver uma clique de disjunção, procede-se da maneira iterativa seguinte: escolhe-se, entre as operações de uma mesma clique, uma operação para executar antes das outras e introduz-se no grafo um

arco ligando a operação escolhida (origem dos arcos) a uma - Regra I - ou a todas as outras operações - Regra II - desta clique (destino dos arcos) e recalcula-se as datas mais_cedo[k] e mais_tarde[k] das operações afetadas por esta modificação. Este procedimento não cria circuitos no grafo, Gondran e Dostatni (1977), de modo que pode-se calcular as datas com a ajuda de uma simples variante do algoritmo de Bellman (1958), para busca dos caminhos mínimos. Numericamente, os grafos estão representados na máquina por listas de antecessores e sucessores dos nós, o que permite um cálculo mais rápido destas datas, Gondran e Minoux (1986).

5.3 Tratamento Dos Movimentos Inter-Células

O seqüenciamento das operações está terminado se a decomposição da fábrica em células não apresentar movimentos inter-células. Se tais movimentos subsistem, é necessário seqüenciá-los, de maneira a tornar realizável a solução obtida na etapa precedente. As disjunções subsistentes a nível de fábrica são resolvidas com a ajuda das Regras I e II utilizadas para seqüenciar cada uma das células: as operações pertencentes à clique da máquina onde está o movimento inter-

célula são submetidas a esta regra e, então, introduz-se arcos que atravessam as fronteiras das células (ou seja, na lista de antecessores e sucessores de um nó, existirão operações executadas em células diferentes). Em seguida, um cálculo de datas ao nível de todo o sistema é efetuado. Para tanto, após o cálculo das datas mais_cedo[k] em cada célula, toma-se o max{ mais_cedo[opω[p]], p = 1 a número de células}, e fixa-se todos os mais _tarde[opω[p]] iguais a este valor.

5.4 Algoritmo para obtenção do seqüenciamento e programação de operações

algoritmo seqüenciamento

P_k = conjunto das operações que precedem a operação k
S_k = conjunto das operações que sucedem a operação k
t[k] = data mais cedo da operação k
T[k] = data mais tarde da operação k
d[k] = duração da operação k
calcular pelo "algoritmo datas": t[k] e T[k]
para toda máquina j **fazer**
    para toda operação k executada na máquina j **fazer**
        marca[k] = (-)
    fim para
    enquanto existir uma operação marcada(-) **fazer**
        aux = k tal que {t[k] + d[k] + T[k]} tenha valor mínimo, [regra II]
        marca[aux]= (+)
        para toda operação k ≠ aux e marcada (-) **fazer**
            $P_k = P_k \cup \{aux\}$
            $S_{aux} = S_{aux} \cup \{k\}$
        fim para
        para toda operação k **fazer**
            calcular pelo "algoritmo datas" : t[k] e T[k]
        fim para
    fim enquanto
fim para
fim algoritmo seqüenciamento

5.5 Algoritmo para obtenção do calendário de fabricação

algoritmo datas

P_k = conjunto das operações que precedem a operação k
S_k = conjunto das operações que sucedem a operação k
t[k] = data mais cedo da operação k
T[k] = data mais tarde da operação k
d[k] = duração da operação k
α = operação fictícia inicial
ω = operação fictícia final
para toda operação k **fazer**
    marca[k] = (-)
fim para
marca[α] = (+)
t[α] = 0
enquanto existir uma operação k marcada (-) com todos k'$\in P_k$ marcados (+) **fazer**
    t[k] = máximo{t[k'] + d[k']}
    marca[k] = (+)
fim enquanto
para toda operação k **fazer**
    marca[k] = (-)
fim para
marca[ω] = (+)
T[ω] = t[ω] [forma clássica]
enquanto existir uma operação k marcada (-) com todos k'$\in S_k$ marcados (+) **fazer**
    T[k] = mínimo{T[k'] + d[k]}
    marca[k] = (+)
fim enquanto
fim algoritmo datas

5.6 Implementação em paralelo

Estrutura da implementação em paralelo usando OCCAM II

"definição dos protocolos de comunicação"
PROTOCOL canal 1
PROTOCOL canal 2

"algoritmo seqüenciamento para a primeira célula"
PROC sequenc1 (canal 1 ca, canal 2 cb)

"algoritmo seqüenciamento para a segunda célula"
PROC sequenc2 (canal 1 ca1, canal 2 cb1)

"algoritmo seqüenciamento para a os movimentos inter-células restantes"
PROC seqintcel (canal 1 ca, ca1, canal2 cb, cb1)

"rotina de entrada e saída de dados do micro PC para Placas transputer
PROC entsai (canal 1 ca, ca1, canal2 cb, cb1)"

"programa principal"
canal 1 ca, ca1
canal 2 cb, cb1
PAR
    entsai (ca, ca1, cb, cb1)
    sequenc1 (ca, cb)
    sequenc2 (ca1, cb1)
SEQ
    seqintcel (ca, ca1, cb, cb1)

Capítulo 6: Experimentos Computacionais

6.1 Exemplo Ilustrativo

No exemplo utilizado (Meguelati, 1988) para ilustrar o método de sequenciamento da produção proposto, dispõe-se de 7 tipos diferentes de máquinas para fabricar 9 tipos de peças. A Tabela 5 fornece os tipos de máquinas disponíveis e os números de máquinas efetivas dentro de cada tipo.

Tabela 5 - Máquinas Disponíveis

Tipo de máquina	Número de equipamentos disponíveis
1	2
2	3
3	4
4	3
5	2
6	2
7	3

A Tabela 6 apresenta, para cada peça a fabricar, o tamanho do lote, o número de operações, o roteiro de fabricação e as durações. Os gráficos de Gantt apresentados foram gerados a partir de uma versão do programa em Turbo Pascal.

Tabela 6 - Problema Inicial

Peça	Tamanho do lote (nº de peças)	Número de operações	Roteiro de fabricação (máquinas)	Duração das operações
1	1	6	1 2 4 5 7 4	1.0 2.0 1.0 3.0 1.0 1.0
2	1	6	1 2 3 1 4 5	1.0 3.0 2.0 1.0 1.0 2.0
3	1	5	1 3 4 5 4	3.0 1.0 2.0 1.0 2.0
4	1	4	2 4 5 6	1.0 2.0 3.0 3.0
5	1	6	2 3 5 6 7 5	1.0 4.0 1.0 2.0 3.0 2.0
6	1	5	1 2 3 5 7	2.0 2.0 3.0 1.0 3.0
7	1	4	3 6 7 6	1.0 1.0 3.0 1.0
8	1	3	2 3 4	4.0 1.0 2.0
9	1	6	1 2 3 5 7 3	2.0 2.0 3.0 2.0 1.0 2.0

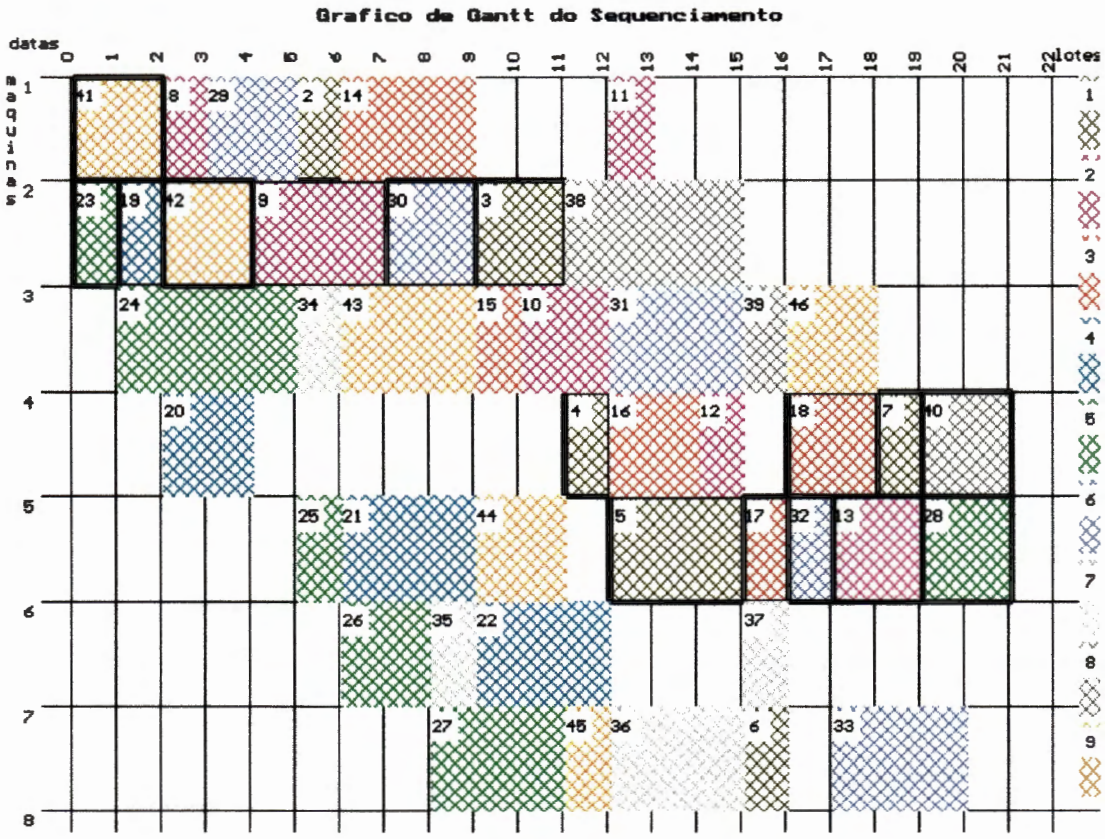

Figura 6 - Gráfico de Gantt para o problema completo

Neste problema, a (I) fase de atribuição já foi realizada pelo método proposto por Ferreira Ribeiro et al. (1996). A (II) fase de organização, pelo mesmo método, pode resultar na criação de diversos arranjos de células. Mostraremos dois destes, o primeiro com zero movimento inter-célula e o outro com dois movimentos inter-células. Ferreira Ribeiro (1991) mostra arranjos deste problema em 2 células, com zero, um, dois, quatro e seis movimentos inter-células.

6.2 Arranjo de duas células e zero movimento inter-célula

A primeira célula de manufatura é formada por 4 peças e 6 máquinas:

Tabela 7 - Célula 4 peças x 6 máquinas

Peça	Tamanho do lote (nº de peças)	Número de operações	Roteiro de fabricação (máquinas)	Duração das operações
4	1	4	2 4 5 6	1.0 2.0 3.0 3.0
5	1	6	2 3 5 6 7 5	1.0 4.0 1.0 2.0 3.0 2.0
7	1	4	3 6 7 6	1.0 1.0 3.0 1.0
8	1	3	2 3 4	4.0 1.0 2.0

A segunda célula de manufatura é formada por 5 peças e 6 máquinas:

Tabela 8 - Célula 5 peças x 6 máquinas

Peça	Tamanho do lote (nº de peças)	Número de operações	Roteiro de fabricação (máquinas)	Duração das operações
1	1	6	1 2 4 5 7 4	1.0 2.0 1.0 3.0 1.0 1.0
2	1	6	1 2 3 1 4 5	1.0 3.0 2.0 1.0 1.0 2.0
3	1	5	1 3 4 5 4	3.0 1.0 2.0 1.0 2.0
6	1	5	1 2 3 5 7	2.0 2.0 3.0 1.0 3.0
9	1	6	1 2 3 5 7 3	2.0 2.0 3.0 2.0 1.0 2.0

A (III) fase do seqüenciamento apresenta os resultados mostrados nas Tabelas 9 e 10 Estes resultados foram obtidos com o auxílio da Regra I no "algoritmo sequenciamento".

Para a primeira célula: A Tabela 9 apresenta a máquina em que cada operação será executada e as datas "mais cedo" e "mais tarde" para o início de execução.

Tabela 9 - Resultados da célula 4x6 (0 movimento inter-célula)

Operação	Lote	Máquina	Mais_cedo	Mais_tarde
1	0	0	0.0	0.0
2	1	2	1.0	6.0
3	1	4	2.0	7.0
4	1	5	6.0	9.0
5	1	6	9.0	12.0
6	2	2	0.0	0.0
7	2	3	1.0	1.0
8	2	5	5.0	5.0
9	2	6	6.0	6.0
10	2	7	8.0	8.0
11	2	5	11.0	13.0
12	3	3	5.0	9.0
13	3	6	8.0	10.0
14	3	7	11.0	11.0
15	3	6	14.0	14.0
16	4	2	1.0	8.0
17	4	3	5.0	12.0
18	4	4	6.0	13.0
19	0	0	15.0	15.0

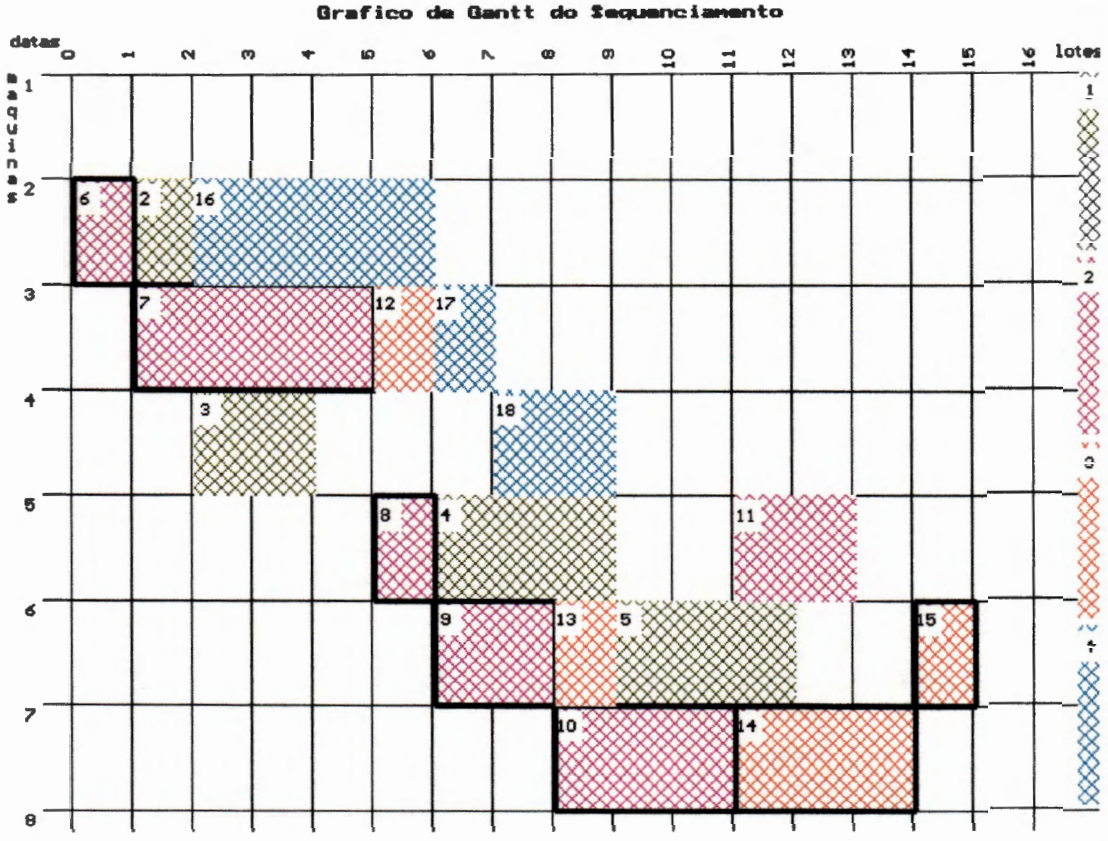

Figura 7 - Gráfico de Gantt para a primeira célula com 0 movimentos inter-células

Para a segunda célula: A Tabela 10 apresenta a máquina em que cada operação será executada e as datas "mais cedo" e "mais tarde" para o início de execução.

Tabela 10 - Resultados da célula 5 x 6 (0 movimento inter-célula)

Operação	Lote	Máquina	Mais_cedo	Mais_tarde
1	0	0	0.0	0.0
2	1	1	5.0	8.0
3	1	2	9.0	9.0
4	1	4	11.0	11.0
5	1	5	12.0	12.0
6	1	7	15.0	16.0
7	1	4	18.0	19.0
8	2	1	2.0	3.0
9	2	2	4.0	4.0
10	2	3	7.0	10.0
11	2	1	9.0	15.0
12	2	4	14.0	16.0
13	2	5	17.0	18.0
14	3	1	6.0	9.0
15	3	3	9.0	12.0
16	3	4	12.0	13.0
17	3	5	15.0	15.0
18	3	4	16.0	17.0
19	4	1	3.0	5.0
20	4	2	7.0	7.0
21	4	3	10.0	13.0
22	4	5	16.0	16.0
23	4	7	17.0	17.0
24	5	1	0.0	0.0
25	5	2	2.0	2.0
26	5	3	4.0	7.0
27	5	5	7.0	10.0
28	5	7	9.0	15.0
29	5	3	13.0	18.0
30	0	0	20.0	20.0

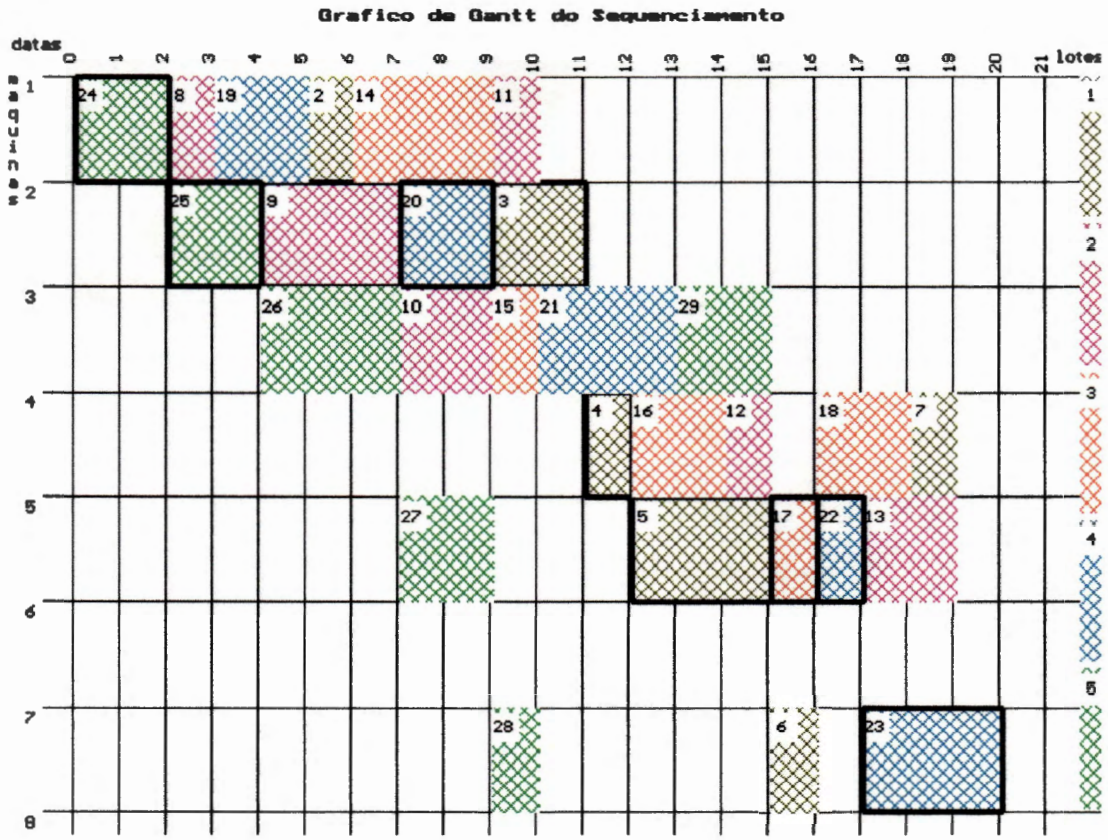

Figura 8 - Gráfico de Gantt para a segunda célula com 0 movimentos inter-células

Como neste caso não se têm movimentos inter-células o procedimento está encerrado. Caso contrário, deveríamos usar o procedimento descrito anteriormente para tratamento dos movimentos inter-células.

Com relação aos tempos obtidos no processamento paralelo, temos os seguintes resultados, obtidos com 2 "*Transputers*" INMOS T8:

Tabela 11 - Desempenho do problema com 0 movimento inter-célula

Problema	N° de processadores utilizados	Valor de mais_cedo[op ω[p]]	tempo em segundos	Speed-Up	Eficiência
Completo	1	15,0	0,417152	---	---
célula 1	1	15,0	0,035264	---	---
célula 2	1	14,0	0,145472	---	---
problema dividido resolvido separadamente	1	15,0	0,180736	2,31	2,31
Problema dividido resolvido concorrentemente	1	15,0	0,199104	2,09	2,09
Problema resolvido em paralelo	2	15,0	0,141376	2,95	1,48

O desempenho da implementação realizada é apresentado na Tabela 11. Nesta Tabela pode ser observado com clareza a eficiência proporcionada pelo paralelismo.

6.3 Arranjo de duas células e dois movimentos inter-células

A primeira célula de manufatura é formada por 5 peças e 6 máquinas:

Tabela 12 - Célula 5 peças x 6 máquinas

Peça	Tamanho do lote (nº de peças)	Número de operações	Roteiro de fabricação (máquinas)	Duração das operações
1	1	6	1 2 4 5 7 4	1.0 2.0 1.0 3.0 1.0 1.0
2	1	6	1 2 3 1 4 5	1.0 3.0 2.0 1.0 1.0 2.0
3	1	5	1 3 4 5 4	3.0 1.0 2.0 1.0 2.0
5	1	6	2 3 5 6 7 5	1.0 4.0 1.0 2.0 3.0 2.0
6	1	5	1 2 3 5 7	2.0 2.0 3.0 1.0 3.0

A máquina do tipo 6 só existe na célula 2. Assim a peça 5, que pertence à célula 1 terá que realizar sua quarta operação na célula 2, caracterizando assim o primeiro movimento inter-célula.

A segunda célula de manufatura é formada por 4 peças e 6 máquinas:

Tabela 13 - Célula com 4 peças x 6 máquinas

Peça	Tamanho do lote (nº de peças)	Número de operações	Roteiro de fabricação (máquinas)	Duração das operações
4	1	4	2 4 5 6	1.0 2.0 3.0 3.0
7	1	4	3 6 7 6	1.0 1.0 3.0 1.0
8	1	3	2 3 4	4.0 1.0 2.0
9	1	6	1 2 3 5 7 3	2.0 2.0 3.0 2.0 1.0 2.0

A máquina do tipo 1 só existe na célula 1. Assim a peça 9, que pertence a célula 2 terá que realizar sua primeira operação na célula 1, caracterizando assim o segundo movimento inter-célula.

A (III) fase do seqüenciamento apresenta os seguintes mostrados na Tabela 14. Para obter estes resultados, utilizou-se a Regra I.

Para a primeira célula de manufatura:

Tabela 14 - Resultados para 5 peças x 6 máquinas (2 movimentos inter-células)

operação	lote	máquina	mais_cedo	mais_tarde
1	0	0	0.0	0.0
2	1	1	5.0	7.0
3	1	2	8.0	8.0
4	1	4	10.0	10.0
5	1	5	11.0	11.0
6	1	7	14.0	16.0
7	1	4	17.0	19.0
8	2	1	2.0	2.0
9	2	2	3.0	3.0
10	2	3	6.0	9.0
11	2	1	9.0	14.0
12	2	4	13.0	15.0
13	2	5	16.0	16.0
14	3	1	6.0	8.0
15	3	3	9.0	11.0
16	3	4	11.0	12.0
17	3	5	14.0	14.0
18	3	4	15.0	17.0
19	5	2	0.0	2.0
20	5	3	1.0	5.0
21	5	5	5.0	10.0
22*	5	6	6.0	11.0
23	5	7	8.0	13.0
24	5	5	18.0	18.0
25	6	1	3.0	4.0
26	6	2	6.0	6.0
27	6	3	10.0	12.0
28	6	5	15.0	15.0
29	6	7	16.0	17.0
30	4	2	0.0	9.0

Para a segunda célula de manufatura:

Tabela 15 - Resultados para 4 peças x 6 máquinas (2 movimentos inter-células)

operação	lote	máquina	mais_cedo	mais_tarde
31	4	4	1.0	10.0
32	4	5	3.0	12.0
33	4	6	8.0	16.0
34	7	3	0.0	9.0
35	7	6	1.0	10.0
36	7	7	2.0	14.0
37	7	6	11.0	19.0
38	8	2	4.0	13.0
39	8	3	8.0	17.0
40	8	4	9.0	18.0
41*	9	1	0.0	0.0
42	9	2	2.0	10.0
43	9	3	4.0	12.0
44	9	5	7.0	15.0
45	9	7	9.0	17.0
46	9	3	10.0	18.0
47	0	0	20.0	20.0

As operações marcadas com (*) são aquelas que geram movimentos inter-células e necessitam da fase algoritmica do tratamento dos movimentos inter-células.

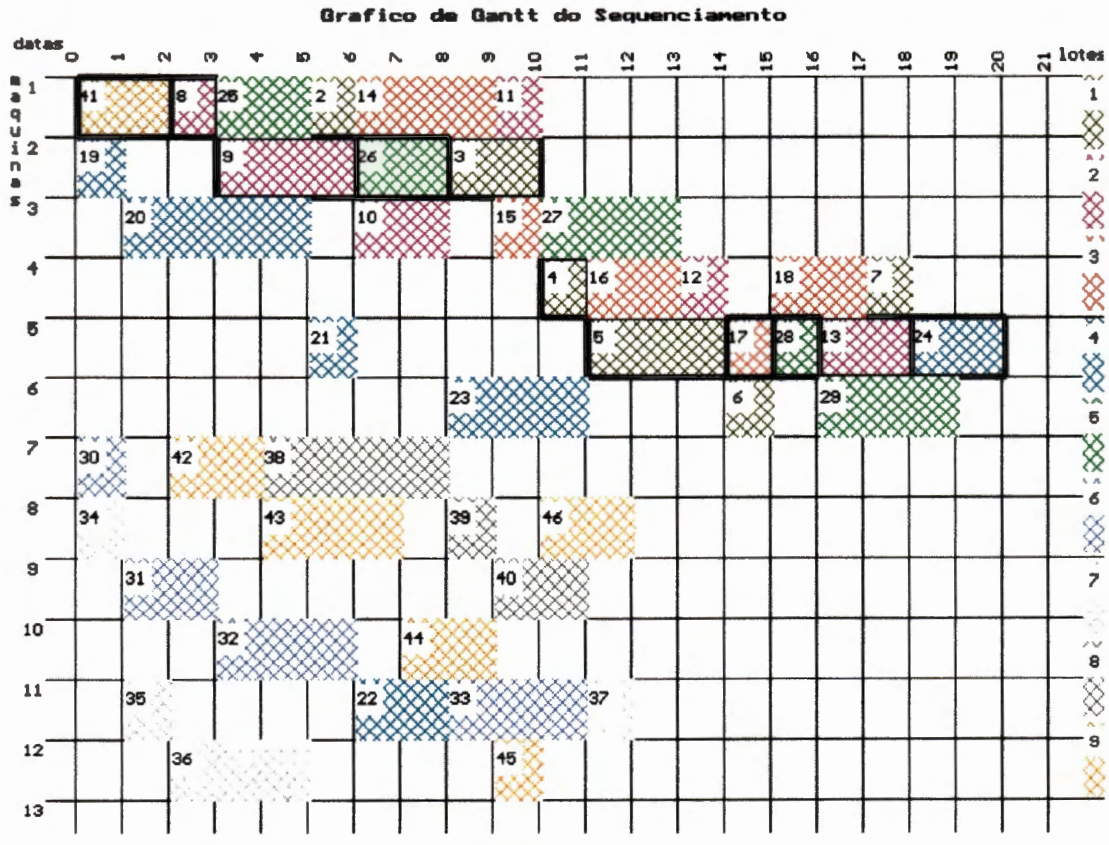

Figura 9 - Gráfico de Gantt para a primeira e a segunda células juntas com 2 movimentos inter-células (operações 22 e 41)

A Tabela 16, abaixo, fornece os tempos de processamento paralelo, obtidos com 2 "*Transputers*" INMOS T8. Estes valores correspondem à execução do programa com os dados e alternativas algoritmicas utilizadas para obter os resultados supra-relatados.

Tabela 16 - Desempenho do problema com 2 movimentos inter-células

Problema	Nº de processadores utilizados	Valor de mais_cedo[op ω[p]]	tempo em segundos	Speed-Up	Eficiência
Completo	1	20,0	0,486712	---	---
célula 1	1	19,0	0,036423	---	---
célula 2	1	20,0	0,167513	---	---
Problema dividido resolvido concorrentemente	1	20,0	0,214579	2,27	2,27
Problema resolvido em paralelo	2	20,0	0,176421	2,76	1,38

O desempenho para o teste computacional considerado é fornecido acima. A eficiência da implementação pode ser avaliada objetivamente.

Capítulo 7 :Conclusão

7.1 Dos Resultados

A proposta apresentada nesta tese consiste em um método paralelo baseado na organização do sistema de produção em células de manufatura para realizar o seqüenciamento de operações. Este método apresenta a vantagem de utilizar uma organização física ou simplesmente lógica da fábrica em células de manufatura. A decomposição do problema global de seqüenciamento em subproblemas de dimensão reduzida, permite uma análise mais simples e administrável do problema. Pode-se encontrar uma solução ótima ou realizável em um tempo de cálculo aceitável, o que permite a utilização do programa correspondente em tempo real.

O programa foi implementado em linguagem OCCAM II sobre um IBM-PC/AT 286 - 12 MHz e dois "Transputers" INMOS T8 com 2 MBytes cada. Cabe ressaltar que neste caso o microcomputador IBM-PC/AT serviu apenas de base para alimentação e base de dados dos programas escritos. Todo o processamento se fez nos processadores das placas INMOS T8.

Seguindo a tendência mundial dos produtos de informática a obsolescência chegou rapidamente sobre estes equipamentos. Entretanto, o método proposto permanece válido seja qual for a linguagem de programação ou plataforma de "hardware" utilizadas. Isto se deve ao fato de que o método tem sua vantagem na implementação em paralelo das operações de cálculo, possível pelo uso de computação paralela, independente do tipo específico dos equipamentos usados neste estudo.

Este método é muito indicado para uso em sistemas de planejamento e controle da produção, especificamente na parte de programação da produção, devido à sua rapidez, mesmo se computadores paralelos não estiverem disponíveis. A programação e principalmente a reprogramação que permite é fundamental para a melhoria do desempenho industrial e capacitação das empresas para enfrentar a concorrência mundial cada vez mais acirrada.

Os tempos obtidos podem ser considerados baixos, mas deve-se ressaltar que o exemplo utilizado, com 9 peças e 12 máquinas (alguns tipos de máquina aparecem em mais de uma célula) de Meguelati (1988), é de pequeno ou médio porte. Os ganhos de tempo obtidos no processo de decomposição em células podem ser perdidos se muitos movimentos

inter-células estiverem presentes na organização da fábrica em células. O uso de processamento paralelo é vantajoso pelo fato do tempo total ser próximo do tempo de cálculo para a célula que demanda o maior tempo de cálculo. Este fato comprova a tese inicial de obter resultados viáveis, em menor tempo, levando-se em conta os movimentos inter-células existentes. Além disso, os tempos de processamento obtidos servem apenas para análises comparativas. Com a evolução dos computadores e das linguagens de programação, resultados considerados excelentes podem ser pulverizados em questão de meses.

Quanto à qualidade das soluções, o algoritmo utilizado obteve a mesma solução encontrada em diversos exemplos da literatura, tais como: Balas (1967), Kaufmann e Labordère (1974), Gondran (1974), Carlier (1978), Meguelati (1988), etc. A proposição de uma maneira "alternativa" para o cálculo das datas permite levar-se em conta as datas de entrega dos produtos acabados, o que não era possível nos algoritmos de Gondran (1974), Gondran e Dostatni (1977) ou Carlier (1978). Assim, pode-se controlar os atrasos na fabricação de certos lotes de peças e definir um seqüenciamento que respeite os prazos de entrega, ao mesmo tempo em que busca o calendário de duração mínima.

7.2 Das Perspectivas de Continuidade

Como perspectivas de continuidade para este trabalho, temos:

1) Paralelização da etapa de seleção e carregamento das máquinas.

   - Resolve-se, para cada coluna j da matriz [lotes de peças x máquinas], um problema de otimização em variáveis bivalentes de grande complexidade. Estes problemas, todavia, são completamente independentes uns dos outros e podem ser resolvidos simultaneamente se temos um computador paralelo disponível.

2) Paralelização da etapa do projeto de células de manufatura.

   - Em um primeiro momento, poder-se-ia implementar em paralelo diferentes algoritmos de partição propostos na literatura e que descrevem resultados de boa qualidade. A solução encontrada que apresentasse o menor número de movimentos inter-células seria a solução aproveitada para seqüenciar e programar as operações de produção.

3) Paralelização de algoritmos específicos deste trabalho.

- Neste caso, cabe um reestudo dos algoritmos utilizados, tais como: programação dinâmica, enumeração implícita, nuvens dinâmicas, etc. com objetivo de executá-los em paralelo e não de utilizá-los da maneira seqüencial em que estão propostos na literatura dentro de uma metodologia paralela mais ampla.

4) Introduzir outros algoritmos de seqüenciamento e programação de operações na metodologia.

- Os algoritmos implementados têm como fundamentos básicos para a resolução das disjunções nas máquinas, a Regra I (Gondran, 1974) e a Regra II (Gondran e Dostatni, 1977). Para o cálculo das datas, tem-se disponível a maneira clássica (Gondran e Minoux, 1986) e a maneira alternativa (Ferreira Ribeiro, 1991). Dada a maior rapidez proporcionada pelo paralelismo, seria interessante testar

algoritmos seqüenciais mais lentos como busca arborescente por separação e avaliação e os diferentes modelos matemáticos de programação inteira.

Referências Bibliográficas

- ACKOFF, R. L., (1975), Planejamento Empresarial, Rio de Janeiro, LTC Ed.
- AMDAHL, G. M., (1967), Validity of the single-processor approach to achieving large-scale computing capabilities, AFIPS Conference Proceedings, 30.AFIPS Press.
- AMINI, M. N., RACER, M., (1994), A Rigourous computacional comparison of alternative solution methods for the generalized assignment problem, Management Science, 40, 7, pp. 623-631.
- ANSHUL, G., GEORGE, K., (1994), Introduction to Parallel Computing Design and Analysis of Algorithms, Benjamin/Cummines.
- ARRUDA, P. E. S., VILA Fº, E. G., (1995), Levantamento do estágio atual de implementação da TG e células de manufatura no Estado de São Paulo, XV ENEGEP / I ICIE, São Carlos, SP, pp. 1559-1562.
- ASHOUR, S., HIREMATH, S. R., (1973), A branch-and-bound approach to the job shop scheduling problem, International Journal of Production Research, 11, 1, pp. 47-58.
- ASKIN, R., SUBRAMANIAN, S., (1987), A cost-based heuristic for GT configuration, Int. J. Prod. Res., 25, 1, pp. 101-114.
- BAKER, K. R., (1974), Introduction to sequencing and scheduling, John Wiley.
- BAKER, K. R., SCHRAGE, L. E., (1978), Finding an optimal sequence by dynamic programming: an extension to precedence-related tasks, Operations Research, 26, 1, pp.111-120.

- BALAS, E., (1967), Discrete programming by filter method, Operation Research, 15, 5, pp. 915-957.
- BALAS, E., (1969), Discrete sequencing via disjunctive graphs: an implicit enumeration algorithm, Oper. Res., 26, pp. 111-120.
- BARBOSA, S. M., FERREIRA RIBEIRO, J. F., (1995), Projeto de células de manufatura com número mínimo de movimentos inter-células, XVIII Cong. Nac. Mat. Apl. Comp., Curitiba, PR, pp. 430-434.
- BELLMAN, R. E. (1958), On a routing problem. Quart. Appl. Math., 16, pp. 87-90.
- BERGE, C., (1973), Graphs and hypergraphs, North-Holland.
- BERTSEKAS, D. P., TSITSIKLIS, J. N., (1989), Parallel and Distributed Computation: Numerical Methods, Prentice-Hall.
- BOUGE, L., (1988), Semantiques du Parallélisme: Un Tour d'Horizont, Rapport de Recherche No 88.6, Laboratoire d'Informatique Fondamentale d'Orléans, França.
- BOWMAN, E. H. (1959), The schedule sequencing problem, Oper. Res., 7, 5, pp 621-624.
- BURBIDGE, J. L., (1975), The introduction of Group Technology, John Wiley.
- CARLIER, J., (1975), Disjonctions dans les ordonnancements, RAIRO, 2, pp. 83-100.
- CARLIER, J., (1978), Ordonnancements à contraintes disjonctives, RAIRO, 12, 4, pp. 333-351.

- CARLIER, J., (1984), Problèmes d'ordonnancement à contraintes de ressources, Tese de Doutorado Universidade Paris VI, França.

- CARLIER, J., CHRETIENNE, P., (1982), Un domaine très ouvert : les problèmes d'ordonnancement, RAIRO, 16, 3, pp. 175-217.

- CARLIER, J., PINSON, E., (1989), A branch-and-bound method for solving the job shop problem, Management Science, 35, 2, pp. 164-176.

- CHAN, H. M., MILNER, D. A., (1981), Direct clustering algorithm for GT in cellular manufacturing, Journal of Manufacturing Systems, 1, 1, pp. 65-74.

- CHANDRASEKHARANM, M. P., RAJAGOPALAN, R., (1987), ZODIAC: an algorithm for concurrent formation of part-families and machine cells, International Journal of Production Research, 25, 6, pp. 835-850.

- CHIAVENATO, I., (1990), Iniciação ao Planejamento e Controle de Produção, MacGraw-Hill.

- DIDAY, E., (1971), Une nouvelle méthode en classification automatique et reconnaissance de formes, Rev. Stat. Appl., 19, 2, pp.213-220.

- DIDRI, N., PROTH, J. M., (1987), Ordonnancement des tâches : une méthode basée sur la Technologie de Groupe, II Conf. Int. Systèmes de Production, Paris, França, pp. 61-74.

- DRUCKER, P., (1997), O admirável mundo do Conhecimento: Uma organização baseada na informação é plana, HSM Management, 1, pp. 63-80.

- DUPONT, L., (1986), Algorithms et ordonnacements, Tese de Doutorado Universidade Joseph Fourier, Grenoble, França.

- EL BAZ, D., (1990), M-Functions and parallel asynchronous algorithms, SIAM J. On Num. Anal., 27, pp. 136-140.

- ERSCHLER, J., ROUBELLAT, F., VERNHES, J. P., (1976), A decision making process for the real time control of a production unit, International Journal of Production Research, 14, 2, pp. 275-284.

- ERSCHLER, J., FONTAN, G., ROUBELLAT, F.,(1979), Potentiels sur un graphe non conjonctif et analyse d'un problème d'ordonnancement à moyens limités, RAIRO, 13, 4, pp. 363-378.

- FERREIRA RIBEIRO, J. F., ALVES, M. R. P. A., (1994), Projeto de células de manufatura por meio de ligações-contrações, XV Cong. Iber. Lat. Amer. Met. Comp. Eng., Belo Horizonte, MG, pp.1470-1476.

- FERREIRA RIBEIRO, J. F., MEDEIROS, C. A., RIBEIRO, C. M., OLIVEIRA, M. M. B., (1996), Seleção e Carregamento de Máquinas, Revista Brasileira de Ciências Mecânicas, 18, 3, pp. 239-247.

- FERREIRA RIBEIRO, J. F., PRADIN, B., (1991), TEGO: un logiciel pour l'ordonnancement d'ateliers de production, GSI'3, Tours, França, pp. 888-896.

- FERREIRA RIBEIRO, J. F., PRADIN, B., (1993). A methodology for cellular manufacturing design. International Journal of Production Research, 31, 1, pp. 225-250.

- FERREIRA RIBEIRO, J. F., RIBEIRO, C. M., (1993), Um método para seqüenciamento baseado na decomposição dos sistemas de manufatura, XXV SBPO, Campinas, SP, pp. 346-350.

- FERREIRA RIBEIRO, J. F., RIBEIRO, C. M., PRADIN, B., (1993), Un algorithme basé sur les îlots de fabrication pour l'ordonnancement, GSI'4, Marseille, França, pp. 219-224.

- FERREIRA RIBEIRO, J. F., (1991), Constituition d'îlots et ordonnancement dans un atelier de production, Tese Doutorado LAAS, Toulouse, França.

- FERREIRA, M. S., RESENDE, M. O., (1995), Um exame à prática do controle de produção em células de manufatura, XV ENEGEP / I ICIE, São Carlos, SP, pp. 1579-1583.

- FERREIRA, R. C., FERREIRA RIBEIRO, J. F., (1995), Coloração de grafos: uma aplicação em TG, XV ENEGEP / I ICIE, São Carlos, SP, pp. 1142-1147.

- FLYNN, M. J., (1966), Very High -Speed computing systems, Proc. IEEE 54, pp. 1901-1909.

- GAREY, M. R., and JOHNSON, D. S., (1979), Computers and Intractability: A Guide to the Theory of NP Completeness, Freeman.

- GLOVER, F., WOOLSEY, E., (1974), Converting the 0/1 polynomial problem to 0/1 linear programming, Oper. Res., 22, pp. 180-182.

- GOMORY, R. E., HU, T. C., (1961), Multi-terminal network flows, Siam J. Appl. Math., 9, 4, pp. 551-570.

- GONDRAN, M., DOSTATNI, M., (1977), Le traitement des exclusives dans Plannec, Bulletin EDF, série C, Mathématiques, Informatique, 1, pp. 69-78.

- GONDRAN, M., MINOUX, M., (1985), Graphes et Algorithmes, Eyrolles.

- GONDRAN, M., MINOUX, M., (1986), Graphs and algorithms, John Wiley.

- GUNASINGH, K. R., LASHKARI, R. S., (1989), Machine grouping problem in cellular manufacturing systems - an integer programming approach, International Journal of Production Research, 27, 9, pp. 1465-1473.

- GUPTA, T., SEIFODDINI, H., (1990), Production data based similarity coefficient for machine-component grouping decisions in the design of a cellular manufacturing system, International Journal of Production Research, 28, 7, 1247-1269

- HILIER, F. S., LIEBERMAN, G.J., (1988), Introdução à Pesquisa Operacional, Campus.

- HIRSH, E., (1990), Les transputers , Eyrolles.

- HOCKNEY, R. W., JESSHOPE, C. R., (1988), Parallel Computers 2: Architecture, Programming and Algorithms, Adam Hilgers, Bristol and Philadelphia.

- HUTCHISON, J., CHANG, Y. L., (1990), Optimal nondelay job shop schedules, International Journal Production Research, 28, pp. 245-257.

- HYER, N. L., WEMMERLÖW, U., (1989), GT in the US manufacturing industry: a survey of current practices, International Journal of Production Research, 27, 8, pp. 1287-1304.

- JOHNSON, S. M., (1954), Optimal two and three stage production schedules with set up times included, Naval Res. Logist Quart., 1, pp 61-68.

- KAUFMANN, A., LABORDERE, A. H., (1974), Méthodes et modèles de la R. O., Dunod.

- KING, J. B., (1980), Machine-component grouping formation in GT: review and extension, Int. J. Prod. Res., 25, 12, pp. 1715-1728.

- KUSIAK, A, CHOW, W. S., (1988) Decomposition of manufacturing Systems, IEEE Journal of Robotics and Automation, 4, 5, pp. 457-471.

- KUSIAK, A., (1987a), The generalized group technology concept, Int. J. Prod. Res., 25, 4, pp. 561-569.

- KUSIAK, A., (1987b), Artificial intelligence and operations research in FMS, Inform. Syst. Oper. Res., 25, pp. 2-12.

- KUSIAK, A., (1987c), Extg-s: a knowledge based system for GT, Int. J. Prod. Res., 26, 5, pp. 887-904.

- LAGEWEG, B. J., LENSTRA, J. K., RINNOOY KAN, A. H. G., (1978), A General Bounding Scheme for the permutation Flowshop Problem, Operations Research, 26, pp. 53-67.

- LASHKARI, R. S., DUTTA, S. P., PADHYE, A. M., (1987), A New Formulation of Operation Allocation Problem in FMS - Math. Modelling and Computac. Exp., International Journal of Production Research, 25, 9, pp. 1267-1283.

- LAVALLEE, I., (1990), Algorithmique Parallèlle et Distribuée ", Hermès, Paris.

- LEE, J. L., VOGT, W. G., MICKLE, M. H., (1982), Calculation of shortest paths by optimal decomposition, IEEE Trans. Syst. Man. Cyb., 12, pp.410-415.

- LIAO, T. W., (1994), Design of Line-Type Cellular Manufacturing Systems for Minimum Operating and Material Handling Costs, International Journal of Production Research, 32, 2, pp. 387-397.

- LOGENDRAN, R., RAMAKRISHNA, P., SRIKANDARAJAH, C., (1994), Tabu-Search-Based Heuristics for FMS in the Presence of Alternative Process Plans, Int. J. Prod. Res., 32, 2, pp. 273-297.

- MANNE, A. S., (1960), On the job-shop scheduling problem, Oper. Res., pp. 219-223.

- MARQUES, E., (1993), Projeto de um elemento de processamento paralelo de um computador maciçamente paralelo para execução dirigida a fluxo de dados, Tese Doutorado USP-EP, São Paulo.

- MAY, D., TAYLOR, R., (1984), OCCAM - an overview, Microprocessors and Microsystems, 8, 2, pp. 73-79.

- Mc MAHON, G., FLORIAN, M. (1975), On scheduling with Ready Times and Due Dates to Minimize Maximum Lateness, Oper. Res., 23, 3, pp. 475-482.

- McCORMICK, W. T., SCHWEITZER, P. J., WHITE, T. W., (1972), Problem decomposition and data reorganization by cluster technique, Oper. Res., 20, 20, pp. 993-1009.

- MEGUELATI, S., (1988), Méthods de classification pour la constituition d'îlots de fabrication et l'ordonnancement, Tese Doutorado INSA, Toulose, France.

- MIELLOU, J. C., (1975), Itérations Chaotiques à Retards - Études de la convergence dans le cas d'espaces partiellement ordonés, CRAS-Paris, série A, t. 280, pp. 233-236.

- MIELLOU, J. C., (1986), Asynchronous iterations and order intervals, Parallel Algorithms and Architectures, North-Holland, pp. 85-96.

- MINOUX, M., (1983), Programation Mathématique - Théorie et Algorithmes (vol. 1 e 2), Dunod.

- MITROFANOV, S. P., (1959), The Scientific Principles of Group Technology, National Lending Library, Londres, Inglaterra.

- MONTEVECHI, J. A. B., (1995), Algoritmo de formação de células de manufatura adaptados à incerteza, XV ENEGEP, I ICIE, São Carlos, SP, pp. 1068-1072.

- MONTEVECHI, J. A. B., GORGULHO JR., J. H. C., (1995), Uma proposta computacional para inferência reversa fuzzy, XV ENEGEP / I ICIE, São Carlos, SP, pp. 1402-1409.

- MUTH, J. F., THOMPSON, G. L., (1963), Industrial Scheduling, Prentice Hall.
- NASR, N., ELSAYED, E.A., (1990), Job shop scheduling with alternative machines, International Journal of Production Research, 28, 9, pp. 1595-1609.
- NEMHAUSER, G. L., WOLSEY, L. A., (1988), Integer and Combinatorial Optimization, John Wiley.
- NEPOMIASTCHY, P., (1978), Résolution d'un problème d'ordonnancement à variables, RAIRO, 12, 3, pp. 249-261
- NORONHA, A. B., FERREIRA RIBEIRO, J. F., (1994a), Um algoritmo branch-and-bound para o seqüenciamento da produção, XIV ENEGEP, João Pessoa, PB, pp 672-677.
- NORONHA, A. B., FERREIRA RIBEIRO, J. F., (1994b), Um método e um programa para o seqüenciamento da produção, XXVI SBPO, Florianópolis, SC, pp 06-11.
- NORONHA, A. B., FERREIRA RIBEIRO, J. F., (1994c), Um método para o seqüenciamento de operações baseado em busca arborescente e proposições, XV CILAMCE, Belo Horizonte, MG, pp. 1452-1459.
- NORONHA, A. B., RIBEIRO, C. M., FERREIRA RIBEIRO, J. F., OLIVEIRA, M. M.B., (1996), Sequenciamento e Programação da Produção, XIX CNMAC, Goiânia (GO), pp. 274-275.
- OLIVEIRA, M. M. B. , FERREIRA RIBEIRO, J. F., RIBEIRO, C. M., (1994a), Seqüenciamento de Operações Baseado na Decomposição dos Sistemas de Manufatura e Uso de Processamento Paralelo, X CBA / VI CLACA, Rio de Janeiro, RJ, pp. 133-138.

- OLIVEIRA, M. M. B. , FERREIRA RIBEIRO, J. F., RIBEIRO, C. M., (1994b), Células de manufatura e processamento paralelo para o seqüenciamento de produção, XV CILAMCE, Belo Horizonte, MG, pp. 1422-1431.

- OLIVEIRA, M. M. B. , FERREIRA RIBEIRO, J. F., RIBEIRO, C. M., (1994c), Seqüenciamento da produção com divisão em cálulas de manufatura e uso de processamento paralelo, XIV ENEGEP, João Pessoa, PB, pp. 678-683.

- OLIVEIRA, M. M. B. , FERREIRA RIBEIRO, J. F., RIBEIRO, C. M., (1996), Scheduling Based on Manufacturing Cells and Parallel Computing, 1996 International Conference of the Swedish Operations Research Association, Lulea University, Lulea, Sweden.

- PAPADIMITRIOU, C. H., STEIGLITZ, K., (1982), Combinatorial Optimization: Algorithms and Complexity, Prentice Hall.

- PIRES, S. R. I., AGOSTINHO, O. L., (1988), Arranjo Físico Celular e sua Influência no Planejamento e Controle de Produção. Anais vol. 1 - VIII ENEGEP (Encontro Nacional de Engenharia de Produção), pp. 96-101.

- PORTMANN, M. C., (1988), Méthodes de décomposition spatiales et temporelles en ordonnancement de la production, RAIRO, 22, pp. 439-451.

- PROTH, J. M., (1986), Group Technology in production management: a tool to simplify some scheduling problems, IEEE Int. Cont. on Robotics and Automation, pp. 1641-1644.

- QUIN, M. J., (1987), Designing Efficient Algorithms for Parallel Computers McGraw-Hill.

- RAJAGOPALAN, R., BATRA, J. L., (1975), Design of cellular production systems - a graph theoretic approach, Int. J. Prod. Res., 13, 6, pp. 567-579.

- RESENDE, M. O., SACOMANO, J. B., (1991), Princípios dos Sistemas de Planejamento e Controle da Produção, EESC-USP.

- RIBEIRO, C. M., (1991), Une méthode parallèle pour le routage dans les réseaux a commutations de paquets, Tese Doutorado LAAS, Toulouse, França.

- RIBEIRO, C. M., EL BAZ, D., (1992), A parallel optimal routing algorithm, Parallel Computing, 18. North-Holland/ Elsevier Sc., pp. 1393-1402.

- RONCONI, D. P., ARMENTANO, V. A., (1993), Um método heurístico baseado em grafos para formação de células de manufatura, Cadernos DEP-UFSCAR, 21, pp. 24-48.

- ROUBELLAT, F., THOMAS, V., (1988), Une méthode et un logiciel pour l'ordonnancement en temps réel d'ateliers, RAIRO, 22, pp. 419-438.

- ROY, B.,(1970), Algèbre moderne et théorie des graphes, Dunod.

- SABOT, G. W., (1995), High Performance Computing: Problem Solving with Parallel and Vector Architectures. Addison-Wesley.

- SAKAROVITCH, M., (1984), Optmisation combinatoire: Graphes et Programmation Linéaire (vol. 1), Programmation Discrète (vol. 2), Hermann.

- SALKIN, H. M., (1975), Integer Programming, Addison Wesley.

- SANKARAN, S., KASILINGAM, R. G., (1993), One Cell Size and Machine Requirements Planning in GT Systems, Eur. J. Op. Res., 69, 3, pp. 373-383.

- SCHENDEL, V., (1984), Introduction to Numerical Methods for Parallel Computers, Ellis Horwood Series, John Wiley and Sons N.Y.

- SÉRIO, L. C., (1990), Tecnologia de Grupo no Planejamento de um Sistema Produtivo, Ícone Editora.

- SHTUB, A., (1989), Modelling Group Technology cell formation as generalized assignment problem. International Journal of Production Research, 27, 5, pp. 775-782.

- SRINIVASAN, G., NARENDAN, T. T. , MAHAVEDAN, B., (1990), An Assignment Model for the Part-Families Problem in GT, Int. J. Prod. Res., 28, 1, pp. 145-152.

- STAROSWIECKI, M., HAPIETTE, M., AMADOU, M., (1991), Optimisation d'une décomposition temporelle d'un ensemble d'ordres de fabrication, III Congrès de Génie Industriel, Tours, France, pp. 907-918.

- SULE, D. R., (1991), Machine Capacity Planning in GT, Int. J. Prod. Res., 29, 9, pp. 1909-1922.

- SZWARCFITER, J.L., (1983), Grafos e Algoritmos Computacionais, Campus.

- TAILLARD, E., (1993), Benchmarks for basic scheduling problems. European Journal of Operations Research, 64, pp. 278-285.

- TSENG, P., BERTSEKAS, D. P., TSITSIKLIS, J. N., (1990), Partially asynchronous parallel algorithms for network flow and other problems, SIAM Control and Optimization, 28, pp. 678-710.

- VANELLI, A., KUMAR, K. R., (1986), A method for finding minimal bottleneck cells for grouping part-machine families, International Journal of Production Research, 24, 2, 387-400.

- VENUGOPAL, V., NARENDRAN, T.T., (1992), Cell formation in manufacturing systems through simulated annealing: An experimental evaluation. European Journal of Operations Research, 63, pp. 409-422.

- VILA Fº, E. G., (1982), Introdução à TG: um novo enfoque em sistemas de produção, Tese de Mestrado USP-EESC-SEM, São Carlos, SP.

- WAGHODEKAR, P. H., SAHU, S., (1984), Machine-component cell formation in group technology: MACE, Int. J. Prod. Res., 22, 6, pp. 937-948.

- WAGNER, H. M., (1959), An integer linear programming model for machine scheduling, Naval Res. Logist Quart., 6, 2, pp 131-140.

- WEI, J. C., GARY, M. K., (1989), Commonality analysis: a linear cell clustering algorithm for group technology, Inter. J. Oper. Res., 27, 12, pp. 2053-2062.

- WILHELM, W. E., SHIN, H. M., (1985), Effectiveness of Alternative Operations in a FMS, Int. J. Prod. Res., 23, 1, pp. 65-79.

- WILLIAMS, S. A., (1990), Programming models for parallel systems, Wiley, Series in Parallel Computing

- WITTE, J., (1980), The use of similarity coeficientes in production flow analyses, Int. J. Prod. Res., 18, 4, pp. 503-514.

- YAMAMOTO, M., (1977), An approximative solution of machine scheduling problems by decomposition method, International Journal of Production Research, 15, 6, pp. 599-608.

- YAMAMOTO, M., NOF, S. Y., (1985), Scheduling/rescheduling in the manufacturing operating system, environment, International Journal of Production Research, 23, 4, pp. 705-722.