Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação

Problema Conjunto de Dimensionamento de Lotes e Programação da Produção

Autor: Claudio Fabiano Motta Toledo
Orientador: Prof. Dr. Paulo Morelato França
Co-orientador: Prof. Dr.Reinaldo Morabitto

Tese de Doutorado apresentada à Faculdade de Engenharia Elétrica e de Computação como parte dos requisitos para obtenção do título de Doutor em Engenharia Elétrica. Área de concentração: **Automação**.

Banca Examinadora

Akebo Yamakami, Prof. Dr. ..... DT/FEEC/Unicamp
Franklina Maria Bragion de Toledo, Prof(a)Dr(a) ..... ICMC/USP
Maria do Socorro Nogueira Rangel, Prof.(a) Dr(a) ..... IBILCE/Unesp
Paulo Morelato França, Prof. Dr. ..... Densis/FEEC/Unicamp
Takaaki Ohishi, Prof. Dr. ..... Densis/FEEC/Unicamp

Campinas, SP
Setembro/2005

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

Toledo, Claudio Fabiano Motta
T575p Problema conjunto de dimensionamento de lotes e programação da produção / Claudio Fabiano Motta Toledo. --Campinas, SP: [s.n.], 2005.

Orientadores: Paulo Morelato França, Reinaldo Morabitto.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Planejamento da produção. 2. Heurística . 3. Bebidas - Industria. 4. Programação (Matemática). I. França, Paulo Morelato. II. Morabitto, Reinaldo. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Titulo em Inglês: Integrated Two -Stage d Lot Sizing and Scheduling Problem.
Palavras -chave em Inglês: Production planning, Heuristic, Drink industry e Programming mathematics.
Área de concentração: Automação
Titulação: Doutor em Engenharia Elétrica
Banca examinadora: Akebo Yamakami, Franklina Maria Bragion de Toledo, Maria do Socorro Nogueira Rangel e Takaaki Ohishi
Data da defesa: 06/09/2005

Resumo

A presente tese de doutorado apresenta, modela matematicamente e soluciona um problema multi-nível de dimensionamento de lotes e programação da produção em um ambiente industrial com máquinas paralelas que apresentam restrições de capacidade, custos e tempos de preparo dependentes da seqüência. O problema é motivado pela realidade encontrada em um setor industrial, em particular o de fabricação e engarrafamento de bebidas. Nesse tipo de indústria a produção envolve dois níveis interdependentes com decisões relativas à armazenagem das matérias-primas e ao engarrafamento das bebidas. As diversas matérias-primas são armazenadas em tanques de onde escoam para as linhas de engarrafamento. O desafio é determinar simultaneamente o dimensionamento e a programação das matérias-primas nos tanques e o envasamento de bebidas nas linhas, onde tempos e custos de trocas dependem do tipo de item previamente armazenado e envasado. O objetivo não foi apenas fornecer uma solução para o problema industrial, mas também estabelecer e solucionar o problema do ponto de vista acadêmico. Um modelo matemático inteiro-misto é proposto com diversas restrições combinadas que até então costumavam ser tratadas separadamente pela literatura. Inicialmente o modelo foi solucionado por meio do pacote GAMS/Cplex. A não existência de testes com modelos similares nos obrigou a criar um conjunto de instâncias para avaliar o modelo e as técnicas de solução desenvolvidas. A solução exata foi viável apenas em instâncias de pequena dimensão devido à complexidade do problema em estudo. Meta-heurísticas foram então propostas e se revelaram como uma alternativa para solucionar instâncias de média e grande dimensão. Os métodos foram capazes de fornecer soluções dentro de um tempo computacional razoável.

Palavras-chave: Programação da produção, Dimensionamento de lotes, Meta-heurísticas, Indústria de bebidas.

Abstract

The present thesis establishes and solves a multi-level lot sizing and scheduling problem with parallel machines and sequence-dependent setup cost and time. The problem was motivated by a real situation found in some industrial settings mainly the soft drink industry. In this kind of industry, the production involves two interdependent levels with decisions about raw material storage and soft drink bottling. The several raw materials are stored in tanks from which they flow to the bottling lines. The challenge is to determine simultaneously the lot sizing and scheduling of raw material in tanks and also in the bottling lines, where setup costs and time depend on the previous items stored and bottled. The objective is not only to provide an industrial problem solution, but also to establish and solve the problem by an academic point of view. Initially, a mathematical model is proposed with several combined constrains that use to be handled apart in the literature. This complex model was solved by the GAMS/Cplex software. The lack of similar models led us to create a set of instances to evaluate the model and the solution techniques developed. The exact model solution was possible only for small-sized instances because of the problem complexity. Therefore, meta-heuristics have been proposed and revealed as the only alternative to solve large instances. These methods have been able to provide solutions with good quality in a reasonable computational time.

Keywords: Scheduling, Lot sizing, Metha-heuristic, Soft drink industry.

À minha avó Esther, à minha tia Shirley, à minha mãe Sônia,
Aos meus irmãos Valdinéia, Anderson, Adriana, Martha e Jonathas.
À minha amada esposa Walkyria.

Agradecimentos

Ao meu orientador, Prof. Paulo Morelato França, agradeço o amplo apoio que ultrapassou o limite da pesquisa principalmente no início deste trabalho.

Ao meu co-orientador, prof. Reinaldo Morabito, agradeço a oportunidade que me foi dada de trabalhar neste tipo de problema e o generoso apoio fornecido por ele durante o desenvolvimento deste trabalho.

Ao Prof. Alf Kimms, agradeço pela ajuda na elaboração do modelo matemático, o apoio irrestrito durante minha estada na *Technische Universität Bergakademie* e, principalmente, pela amizade surgida durante e após minha permanência na Alemanha.

À Fapesp, pelo apoio financeiro.

Aos colegas de doutorado Vinícius Jacques, Alexandre Mendes, Luciana Buriol, Marcelo Ventura e Denise Sodero pelo apoio.

Aos funcionários da FEEC, em especial à Noêmia, Márcia, Maria José e Rose (BAE).

À minha tia Shirley pelos sacrifícios pessoais realizados em benefício dos seus sobrinhos.

À minha mãe Sônia por me deixar partir ainda jovem para seguir meu destino.

Aos meus irmãos Valdinéia, Anderson e Adriana agradeço pelo amor incondicional e pela fé em mim depositada.

À minha avó Esther por ensinar aos netos a importância do conhecimento e dar a eles a chance de mudarem sua realidade através da educação.

À minha filha Bianca pela paciência e compreensão.

À minha esposa Walkyria pelo incansável e abnegado apoio durante todos esses anos de trabalho. Não imagino como seria possível essa caminhada sem você ao meu lado. Te amo!

Sumário

Lista de Figuras

Lista de Tabelas

Abreviaturas

PCDLPP	Problema Conjunto de Dimensionamento de Lotes e Programação da Produção
QEP	Quantidade Econômica do Pedido
PLE	Problema do Lote Econômico
CC	Ciclo Comum
PDLC	Problema de Dimensionamento de Lote Capacitado
PDDPL	Problema Discreto de Dimensionamento e Programação dos Lotes
PCDL	Problema Contínuo de Dimensionamento de Lote
PPDPL	Problema Proporcional de Dimensionamento e Programação dos Lotes
PGDPL	Problema Geral de Dimensionamento e Programação de Lote
PFR	Problema de Fluxo em Rede
PPIM	Problema de Programação Inteira Mista
AG	Algoritmos Genéticos
AM	Algoritmos Meméticos
BL	Busca Local
SA	*Simulated Annealing*
TA	*Threshold Accepting*
Desv	Desvio
Desv Méd	Desvio Médio
Desv Máx	Desvio Máximo
Desv Mín	Desvio Mínimo
HFV	Heuristica de Fixação de Variáveis

Capítulo 1

Descrição do Problema

1.1 Introdução

A motivação para esta tese nasceu de problemas industriais envolvendo a fabricação de bebidas. Para efeito de melhor compreesão, o problema é ambientado no setor de bebidas e usa nomes e nomenclatura deste tipo de indústria (tanques, xaropes, etc.). A produção em geral é separada em dois níveis ou estágios. No primeiro nível, as matérias-primas são armazenadas em tanques de onde escoam para diversas linhas de produção ou engarrafamento. No segundo nível, as linhas de produção realizam o envase da matéria-prima em embalagens de diferentes tamanhos resultando na bebida ou produto final. Essas embalagens podem ser garrafas, latas, galões do tipo *bag-in-boxes* entre outras. O problema consiste em estabelecer simultaneamente o dimensionamento e a programação de matérias-primas e bebidas, respectivamente, nos tanques e nas linhas de produção. O objetivo principal deste capítulo é apresentar os dois níveis de produção, as restrições existentes em cada um deles e o enquadramento do problema como um problema de dimensionamento e programação da produção.

1.2 Níveis de produção do problema

A fabricação de bebidas envolve o uso de diferentes matérias-primas que misturadas a outros componentes resultam nos xaropes. Os xaropes ficam armazenados em tanques de onde escoam para as linhas de produção. A combinação do tipo de xarope que escoa dos tanques com o tipo de envase realizado nas linhas de produção resulta na bebida ou produto final. Vamos considerar que as demandas por bebidas são planejadas em um horizonte de tempo mensal e devem ser atendidas ao final de cada semana de trabalho.

O problema reside no planejamento e na programação da produção de bebidas fabricadas em

plantas que, num primeiro nível, manipulam matérias-primas (xaropes) para produzir diferentes tipos de produtos (bebidas) que, num nível subseqüênte, devem ser embalados/envasados em linhas de produção dispostas em paralelo. No primeiro nível, o problema envolve o dimensionamento e a programação dos xaropes. Deve-se decidir quanto e quando determinado xarope será armazenado em um dos tanques disponíveis. No segundo nível, o problema envolve o dimensionamento e a programação das bebidas ou produtos finais. Deve-se decidir quanto e quando determinado produto será colocado em produção nas linhas de engarrafamento. A figura 1.1 abaixo apresenta uma visão geral do processo de produção.

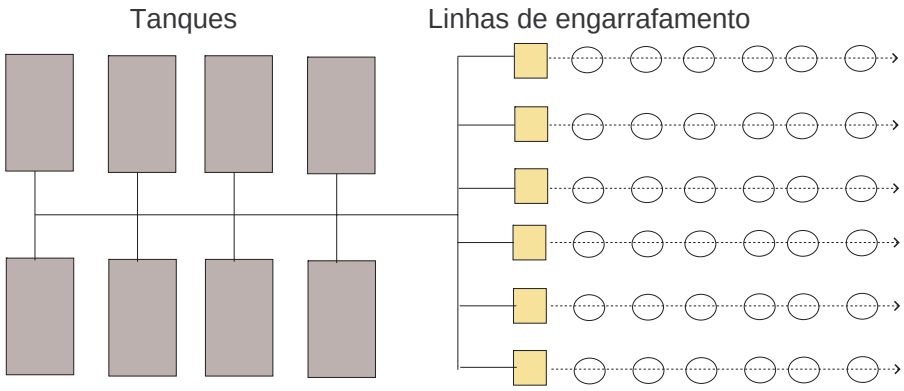

Fig. 1.1: Níveis de produção.

Observe que nos dois níveis as linhas e tanques podem ser vistos como processadores em paralelo. As decisões relativas aos tanques repercutem nas decisões tomadas para as linhas de engarrafamento e vice-versa. Por exemplo, se determinado produto está em produção nas linhas, o xarope utilizado deverá estar armazenado em um tanque. Temos uma interdependência entre os dois níveis que deve ser levada em conta durante o processo de tomada de decisões.

1.3 Restrições e Decisões Envolvidas

As restrições e decisões existentes no problema industrial serão esclarecidas nesta seção. Inicialmente, apresentaremos as restrições relativas aos tanques:

1. Todos os tanques envolvidos têm uma capacidade máxima de armazenagem que não poderá ser ultrapassada. Os tanques também devem ser abastecidos com uma quantidade mínima de xarope.

2. Cada tanque pode armazenar um único xarope por vez, mas o mesmo xarope poderá estar armazenado em diversos tanques.

3. Um tanque será reabastecido apenas quando o xarope nele armazenado for completamente escoado para as linhas de produção.

4. Existe um custo relacionado à troca de um xarope por outro nos tanques. Também existe um gasto de tempo durante esta troca. Isto ocorre porque o armazenamento de um novo xarope no tanque exige a sua limpeza. Tanto o tempo quanto o custo de troca dependem do produto que estava previamente no tanque.

As decisões relativas aos tanques envolvem:

1. O tipo e a quantidade de xarope que será armazenado em cada tanque

2. A seqüência em que os xaropes são produzidos e armazenados em cada tanque.

As restrições relativas às linhas de produção são:

1. As linhas podem realizar mais de um tipo de envase. Por exemplo, uma mesma linha pode ser ajustada para engarrafar o xarope em uma embalagem de $600ml$, $1l$ ou $2l$ dependendo da programação estabelecida.

2. Se um novo xarope é escoado, a linha deverá ser preparada para recebê-lo. O tempo e o custo envolvido no preparo da linha dependem do tipo de bebida previamente produzida.

3. O tempo de processamento depende de cada linha e do tipo de envase realizado. Por exemplo, duas linhas podem realizar o mesmo tipo de envase em diferentes velocidades. Uma mesma linha em geral tem diferentes tipos de tempo de processamento para diferentes tipos de envase.

4. Trocas ou reposições de xarope em um tanque exigem a interrupção da produção nas linhas que o utilizam.

5. A demanda semanal deve ser atendida e o excedente produzido será estocado.

As decisões relativas às linhas de produção envolvem:

1. A definição do tipo e da quantidade de bebida que será produzida em cada linha.

2. A seqüência em que os produtos serão produzidos pelas linhas.

As decisões e as restrições relativas aos tanques e às linhas de engarrafamento são interdependentes, pois:

1. As quantidades de xaropes armazenados nos tanques devem ser suficientes para atender a produção nas linhas.

2. A programação dos xaropes nos tanques deve estar sincronizada à programação das linhas. Não poderá ocorrer o envase de determinada bebida, se o xarope correspondente não estiver armazenado em um tanque.

1.4 Classificação do Problema

O problema industrial considera dois níveis interdependentes. Os xaropes programados nos tanques devem atender as demandas das linhas de produção e vice-versa. O escoamento do xarope programado nos tanques é utilizado na fabricação dos produtos finais nas linhas. Isso caracteriza o problema industrial como sendo multi-nível.

Uma das decisões a ser tomada é a definição da quantidade de cada xarope a ser armazenada nos tanques e a definição da quantidade de cada produto a ser produzido nas linhas. Logo, o problema procura determinar o dimensionamento de lotes nos dois níveis de produção. Além disso, para cada lote produzido nas linhas, uma seqüência de produção deve ser estabelecida. Da mesma forma, a seqüência em que os tanques serão ocupados por cada lote de xarope deve ser estabelecida. Temos um problema de programação de lotes tanto nas linhas quanto nos tanques.

O preparo das linhas(tanques) para produzir(armazenar) um novo produto(xarope) ocasiona um gasto de tempo e um custo relativo à interrupção da produção. Além disso, o gasto de tempo e o custo envolvido dependem do tipo de produto ou xarope previamente produzido na linha ou armazenado no tanque. Desta forma, o problema proposto também é um problema de programação da produção com custos e tempos de preparo dependentes da seqüência nos dois níveis de produção.

Tanto os tanques como as linhas podem ser vistos como máquinas (ou processadores) dispostas em paralelo que processam xaropes e produtos, respectivamente. Portanto, trata-se de um problema de máquinas paralelas nos dois níveis.

O problema apresenta restrições de capacidade. As demandas devem ser atendidas ao final de cada semana. As linhas têm diferentes velocidades de processamento que limitam a capacidade de produção dentro dos períodos considerados. Além disso, os tanques possuem uma capacidade máxima de armazenagem que limita a quantidade de xarope disponível à produção. A combinação desses fatores restringe a capacidade produtiva da planta industrial.

Isto posto, o problema pode ser classificado como um problema multi-nível de dimensionamento de lote e programação da produção em máquinas paralelas com restrições de capacidade, custos e tempos de preparo dependentes da seqüência. Iremos nos referir a tal problema simplesmente como um Problema Conjunto de Dimensionamento de Lotes e Programação da Produção (PCDLPP) no decorrer deste trabalho.

1.5 Conclusão

O presente capítulo apresentou em maiores detalhes o problema proposto através de sua ambientação como um problema de produção de bebidas. As restrições existentes e as decisões a serem tomadas nos levaram a classificá-lo como um problema multi-nível de dimensionamento de lote e programação da produção em máquinas paralelas com restrições de capacidade, custos e tempos de preparo dependentes da seqüência. Trataremos tal problema simplesmente por Problema Conjunto de Dimensionamento de Lotes e Programação da Produção (PCDLPP). Conforme é visto no Capítulo 2, não há muitos trabalhos considerando simultaneamente custos e tempos de preparo dependentes da seqüência em problemas de dimensionamento e programação da produção. Os trabalhos se tornam mais escassos quando máquinas paralelas são consideradas. Além disso, o aspecto multi-nível do problema impõe uma sincronia e uma interdependência na programação realizada nos dois níveis de produção. Todas essas particularidades nos levam a concluir que estamos propondo nesta tese um problema complexo e pouco estudado na área de dimensionamento e programação da produção.

Capítulo 2

Revisão Bibliográfica

2.1 Introdução

A presente tese de doutorado estuda o Problema Conjunto de Dimensionamento de Lotes e Programação da Produção (PCDLPP). Existe uma ampla literatura que aborda separadamente o Problema de Dimensionamento de Lotes (Bahl et al. (1987) e Kuik et al. (1994)) e o Problema de Programação da Produção (Baker (1974), Pinedo (1995) e Blazewicz et al. (1996)). O PCDLPP foi classificado no capítulo 1 como sendo um problema multi-nível de dimensionamento de lote e programação da produção em máquinas paralela com restrições de capacidade, custos e tempos de preparo dependentes da seqüência. Esta revisão bibliográfica irá concentrar-se em publicações que apresentem problemas e modelos com aspectos comuns ao PCDLPP, principalmente os trabalhos que nos conduziram ao modelo matemático proposto no capítulo 3 e aos métodos de resolução do capítulo 4.

2.2 Primeiros Modelos

O primeiro modelo envolvendo o problema de dimensionamento de lote é o chamado Quantidade Econômica do Pedido (QEP). O QEP considera o processo de produção em um único nível (nível-simples), opondo-se a processos com várias etapas intermediárias e interdependentes (multi-nível). O processo de produção não tem restrições de capacidade, portanto, sempre existirá recursos para o processamento de cada item (problema de item-simples). As demandas ocorrem todo o tempo a uma taxa constante (demandas estacionárias). O horizonte de planejamento é ilimitado e a solução ótima analítica não é difícil de ser obtida (Wagner and Whitin (1958), Erlenkotter and Harris (1990), Harris (1990) e Wagelmans et al. (1992)).

A partir do QEP, outros modelos foram estabelecidos com hipóteses que os aproximavam de situações reais. O Problema do Lote Econômico (PLE) é um deles, onde a capacidade disponível

passa a ser limitada. Os recursos disponíveis precisam ser compartilhados caracterizando o PLE como um problema multi-item, mas ainda é um modelo contínuo no tempo com demandas estacionárias em um horizonte de planejamento infinito (Elmaghraby (1978)). A resolução ótima deste problema é NP-difícil (Hsu (1983) e Davis (1990)) tornando comum o uso de métodos heurísticos (Dobson (1987) e Zipkin (1991)).

Um modelo para o PLE com máquinas paralelas é estabelecido por Carreno (1990). As máquinas são idênticas em relação ao custo e à taxa de produção. Não há interferência entre os itens e suas demandas devem sempre ser atendidas. Um item não poderá ser produzido em mais de um processador e somente será produzido quando seu estoque for zero. O objetivo é minimizar o custo médio da produção e do estoque. Carreno utiliza uma abordagem chamada Ciclo Comum (CC) para solucionar o PLE com máquinas paralelas. Um ciclo de tempo, definido como o intervalo de tempo entre sucessivas produções de um dado item e comum a todos os itens, é calculado. Um procedimento heurístico utilizando a abordagem CC é desenvolvido e aplicado a instâncias com 10 máquinas e 100 produtos. Neste procedimento, o problema é decomposto em duas partes. Primeiro, determina-se a alocação dos itens às máquinas com o ciclo comum de tempo dado. Em seguida, calcula-se um novo ciclo comum de tempo que minimize os custos considerando-se a alocação anteriormente determinada.

Outros problemas trabalham tanto com horizonte de planejameno finito como com capacidade disponível limitada. Há modelos para estes problemas nos quais o tamanho e a capaciadade disponível no período são suficientes para que se produzam vários itens. Por outro lado, há modelos onde os períodos e a capacidade disponível são suficientes para se produzir um ou no máximo dois itens. Esses modelos serão analisados nas próximas seções.

2.3 Problema de Dimensionamento de Lote Capacitado (PDLC)

Uma formulação clássica para o PDLC é estabelecida com as seguintes hipóteses:

- $j = 1, ..., J$ itens devem ser produzidos em uma única máquina.
- O horizonte de planejamento (T) é finito e dividido em vários períodos $t = 1, ..., T$.
- C_t é a capacidade da máquina no período t.
- p_j é o consumo de recurso necessário para produzir uma unidade do item j.
- $d_{j,t}$ é a demanda pelo item j que deverá ser satisfeita no período t, sem atrasos.
- s_j é o custo de preparação incorrido ao ajustar a máquina para o item j.
- h_j é o custo por unidade de item j que permanece em estoque ao final de cada período.

- $x_{j,t}$ é a variável que indica a quantidade do item j produzida no período t.
- $I_{j,t}$ é a variável que indica a quantidade do item j em estoque ao final do período t.
- $z_{j,t}$ é a variável que indica se ocorrerá a produção do item j no período t ($z_{i,t} = 1$) ou não ($z_{i,t} = 0$).
- Hipótese básica do modelo: um custo de preparação ocorrerá para cada lote produzido em um período.

A seguinte formulação matemática representa o PDLC:

$$\textit{Minimizar} \sum_{j} \sum_{t} (s_j z_{j,t} + h_j I_{j,t}) \tag{2.1}$$

$$I_{j,t-1} + x_{j,t} - I_{j,t} = d_{j,t} \qquad \forall j, t \tag{2.2}$$

$$\sum_{j} p_j x_{j,t} \leq C_t; \qquad \forall t \tag{2.3}$$

$$p_j x_{j,t} \leq C_t z_{j,t} \qquad \forall j, t \tag{2.4}$$

$$z_{j,t} \in \{0, 1\} \qquad \forall j, t \tag{2.5}$$

$$I_{j,t}, x_{j,t} \geq 0 \qquad \forall j, t \tag{2.6}$$

A função objetivo (2.1) minimiza a soma dos custos de preparação e custos de estoque. A restrição (2.2) indica o balanceamento de estoque de cada item j. A restrição (2.3) assegura que a capacidade disponível na máquina seja respeitada, ou seja, a produção estabelecida no período t não excederá a capacidade C_t da máquina. A restrição (2.4) garante que para haver produção do item j ($x_{j,t} > 0$), deve ocorrer uma atribuição do item à máquina ($z_{j,t} = 1$). Logo, o correspondente custo de preparação será incorrido conforme estabele a hipótese básica do problema. A solução do problema determina qual item e em qual quantidade será produzido nos diversos períodos. Aspectos relativos ao seqüênciamento dos itens em t não estão incluídos no modelo.

A respeito da complexidade deste problema, Bitran and Yanasse (1982) provam que a resolução ótima deste problema é NP-difícil. Métodos de resolução ótima para o PDLC são apresentados em Barany et al. (1984), Chen and Thizy (1990) e G.D.Eppen and Martin (1987). A complexidade do problema faz com que métodos heuristicos sejam bastante aplicados (Maes and van Wassenhove (1988), Maes et al. (1991), Kuik et al. (1993), Kirca and Kökten (1994), Hindi (1996) e Özdamar and Barbarosoglu (2000)).

Modelos e soluções para o PDLC considerando custos e tempos de preparo podem ser encontrados nos trabalhos de Trigeiro et al. (1989), Haase (1993), Haase (1996), Armentano et al. (1999), Özdamar

and Bozyel (2000) e Hindi et al. (2003). Haase and Kimms (2000) apresentam o PDLC com custo e tempo de preparação dependentes da seqüência. Decisões de programação e dimensionamento de lotes são tratadas conjuntamente no modelo estabelecido para este problema. As seguintes hipóteses foram consideradas:

- Se a produção de um item começa em um período, então seu estoque precisa ser zero ao final do período anterior.
- Cada atribuição de um item para uma máquina ocorre dentro do período. Assim, o item produzido ao final do período t estará disponível na máquina no início do período $t+1$.
- O lote de um item não poderá ser dividido dentro de cada período. Por exemplo, uma seqüência de produção (j_1, j_2, j_1) em t não é permitida.

O método de resolução adotado é do tipo *branch and bound*. Para cada período $t = T, T-1, ..., 1$; um conjunto SEQ_t com todas as seqüências (combinações) possíveis de itens é estabelecido. Assim, para $J = 2$ teríamos: SEQ_t= {(j_1), (j_1, j_2), (j_2),(j_2, j_1)}. Chama-se seqüência possível aquela que não viola as restrições de capacidade. Uma seqüência $seq_t \in SEQ_t$ é selecionada aleatóriamente (passo *branch*). O método irá "podar"a árvore de busca (passo *bound*) segundo dois critérios:

1. Se a capacidade remanescente da máquina nos demais períodos ($t-1, t-2, ...2, 1$) é capaz de atender as demandas deste períodos e as demandas remanescentes ($\sum_{j=1}^{J} \sum_{\tau=t}^{T} (d_{j,t} - x_{j,t})$).
2. Se o custo de seq_t atende aos limites inferior e superior estabelecidos pelo método.

Os autores solucionam instâncias com $J = 2, 3, 4, ..., 10$ produtos e $T = 3, 4, 5, ..., 10, 15, 20$ períodos. Diversos parâmetros do problema são gerados aleatóriamente, mas os custos de preparação (custos de troca) são proporcionais ao tempo de preparação:

$$sc_{i,j} = f_{sc} st_{i,j}, i, j = 1, ..., J$$

onde, $sc_{i,j}$ é o custo de preparação do item i para o item j, $st_{i,j}$ é o tempo de preparação e o parâmetro f_{sc} assume os valores de 50 e 500 nos testes computacionais realizados. Soluções ótimas foram alcançadas na maioria das instâncias e o tempo computacional mostrou-se sensível ao crescimento dos parâmetros J e T.

Gupta and Magnusson (2005) também propõem uma formulação para o PDLC com custos e tempos de preparo dependentes da seqüência. Os autores consideram que o preparo da máquina é mantido de um período para o outro e durante períodos ociosos. Instâncias de elevada dimensão são solucionadas por um método heuristico. A heurística estabelece inicialmente uma solução factível para em seguida executar passos de melhoria dessa solução. Alguns limitantes inferiores são determinados utilizando versões relaxadas do modelo original. Esses modelos são obtidos através de agregação de

linhas (restrições) e relaxação de variáveis binárias que passam a ser contínuas. Os modelos relaxados são solucionados pelo *solver* Cplex.

Özdamar and Birbil (1998) propõem um modelo geral para o PDLC considerando máquinas paralelas, diversos tipos de custos (estoque, preparação, processamento e tempo extra) e uma produção multi-estágio dos itens. Todavia, o autor soluciona um modelo simplificado com as seguintes características:

- As máquinas paralelas apresentam diferentes tempos de processamento para cada item (não-relacionadas).
- Os lotes não podem ser divididos entre as máquinas.
- Não há custo de preparação envolvido mas há tempo de preparação.
- A produção dos itens é do tipo estágio-simples.
- Admite a existência de uma capacidade extra disponível (tempo extra) em cada máquina. A utilização deste recurso gera um custo (custo do tempo extra).

Os seguinte parâmetros e variáveis de decisão são consideradas:

- co: custo por hora extra utilizada.
- $CO_{k,t}$: capacidade extra da máquina k no período t.
- $st_{j,k}$: Tempo de preparação gasto pelo item j na máquina k.
- p_{jk}: consumo de recurso necessário para produzir uma unidade do item j na máquina k.
- L_k: conjunto de itens produzidos pela máquina k.
- F_j: conjunto de máquinas capazes de produzir o item j.
- M: número muito grande.
- $C_{k,t}$: capacidade disponível na máquina k no período t.
- $O_{k,t}$: quantidade de tempo extra gasto na máquina k no período t.
- $x_{j,k,t}$: quantidade do item j produzido na máquina k no período t.
- $z_{j,k,t}$: variável binária que indica se o item j será produzido na máquina k no período t.

A formulação matemática do problema é a seguinte:

$$\textit{Minimizar} \sum_{j,t} h_j I_{j,t} + \sum_{t,k} coO_{k,t} \tag{2.7}$$

$$I_{j,t-1} + \sum_{k \in F_j} x_{j,k,t} - I_{jt} = d_{jt} \qquad \forall j,t \tag{2.8}$$

$$\sum_{j \in L_k} p_{j,k} x_{j,k,t} + st_{j,k} z_{j,k,t} \leq (C_{k,t} + O_{k,t}); \qquad \forall k,t \tag{2.9}$$

$$O_{k,t} \leq CO_{k,t} \qquad \forall k,t \tag{2.10}$$

$$\sum_{k \in F_j} z_{j,k,t} \leq 1 \qquad \forall j,t \tag{2.11}$$

$$x_{j,k,t} \leq M z_{j,k,t} \qquad \forall j,k,t \tag{2.12}$$

$$x_{j,k,t} \geq 0, z_{j,k,t} \in \{0,1\} \qquad \forall k,t,j \tag{2.13}$$

A função objetivo (2.7) minimiza o custo de estoque e o custo do tempo extra utilizado. A restrição (2.9) assegura que o tempo de processamento e preparação não ultrapassem a capacidade do período mais o tempo extra utilizado. A restrição (2.10) determina um limite superior para o tempo extra utilizado por cada máquina em cada peíodo.

A programação e o dimensionamento de lotes são tratados conjuntamente. O método de resolução utiliza Algoritmo Genético (AG) associado a uma busca local híbrida. Esta busca local envolve uma busca tabu e *simulated annealing*. Destacamos a seguir alguns aspectos do AG:

- **Codificação do cromossomo**: Um cromossomo (solução) c representará a variável $x_{j,k,t}$ em duas estruturas,

  1. $q_{c,j,t}$: tamanho do lote do item j no período t na solução c.;
  2. $a_{c,j,t}$: máquina na qual o lote do item j será produzido em t.

- **Operador de *crossover***: Um *crossover* do tipo dois pontos é utilizado. Dois cromossomos pais e dois períodos ($t1$ e $t2$) são escolhidos aleatóriamente. Para todos os itens j com $t1 \leq t \leq t2$, as estruturas $q_{1,j,t}(a_{1,j,t})$ e $q_{2,j,t}(a_{2,j,t})$ são trocadas entre os cromossomos pais. O *crossover* pode gerar cromossomos filhos que violam a equação de balanceamento de estoque. Para evitar isto, é executado um procedimento do tipo *backward* e outro do tipo *forward* encarregados de percorrer os cromossomos filhos corrigindo eventuais faltas ou sobras no atendimento das demandas.

Os procedimentos de ajuste do tipo *backward* e *forward* restauram apenas as infactibilidades relacionadas ao estoque. Um cromossomo filho poderá ser factível em relação às restrições de estoque

mas infactível em termos de capacidade. As infactibilidades relacionadas à capacidade disponível são penalizadas. Assim, o AG realiza sua busca dentro de um espaço composto também por soluções infactíveis. O *crossover* é elitista, ou seja, apenas cromossomos filhos com melhor valor de *fitness* são aceitos e ocupam o lugar dos piores indivíduos na população.

Quando o AG converge para uma mesma área do espaço de soluções, uma busca local é executada. Essa busca atua sobre alguns cromossomos selecionados aleatóriamente. As vizinhanças de um cromossomo são alcançadas através de movimentos que aumentam e reduzem lotes de itens de uma mesma família (mesmo tipo) em períodos diferentes. Se esse aumento e redução ocorre dentro de um mesmo período, a quantidade do lote permanecerá a mesma e os itens retirados de um máquina são inseridos necessariamente em uma outra máquina. O procedimento combina Busca Tabu com *Simulated Annealing* pois mantém uma lista de movimentos previamente realizados e um esquema de aceitação ou recusa de movimentos não promissores.

Os testes computacionais são realizados em dois conjuntos de problemas. O primeiro é composto por 5 itens, 3 máquinas e 4 períodos. O segundo tem 20 itens, 5 máquinas e 6 períodos. Um total de 108 instâncias são geradas para o primeiro problema e 36 para o segundo. A abordagem utilizada origina três métodos. O primeiro considera que toda a população inicial é gerada aleatóriamente (chamado GATA1), o segundo método (GATA2) considera que parte da população inicial é gerada utilizando-se o método híbrido de busca local. Neste caso, a busca local gera soluções factíveis. O terceiro método (GATA3) trabalha com subpopulações, realizando cruzamento entre os indivíduos destas subpopulações. Os resultados indicam um melhor desempenho de GATA3 nos dois conjuntos de instâncias, apesar de levar mais tempo do que GATA1 que também obtém bons resultados. O uso de parte da população inicial factível levou a uma convergência prematura de GATA2 nos dois conjuntos de instâncias. Esta convergência prematura levou GATA2 a um desempenho inferior aos demais métodos.

Kang et al. (1999) apresentam o PDLC com máquinas paralelas e custos de preparação dependentes da seqüência dos itens. O problema é modelado considerando-se um conjunto de subseqüências de produção. A união destas subseqüências estabelece a programação da produção de uma máquina durante todo o horizonte de planejamento. Assim, a programação da produção é dividida em seqüências de produção por período. Por sua vez, cada período apresentará um número máximo de seqüências-divididas (*splitting-sequences*). Esta abordagem leva à resolução de subproblemas mais tratáveis do que o original. O método de resolução desenvolvido baseia-se em geração de colunas e *branch & bound*. Uma heurística de arredondamento é utilizada para avaliar os nós gerados pela árvore de busca. Outra heurística de melhoramento realiza uma busca em vizinhança nas soluções. Tanto a formulação do problema quanto os métodos de resolução foram desenvolvido para solucionar as instâncias CHES (Baker and Muckstadt (1989)). Essas instâncias apresentam dados de problemas

fornecidos pela Dupont, BASF, James River e Champion International. Os autores conseguem obter bons resultados quando comparados aos existentes até então para estas instâncias.

Clark and Clark (2000) também propõem e solucionam um PDLC com máquinas paralelas, mas considerando tempos de preparo dependentes da seqüência. O modelo exato proposto pelo autor é muito complexo para solucionar instâncias de elevada dimensão. Por isso, eles estabelecem modelos relaxados que são solucionados utilizando abordagens baseadas em horizonte rolante e fixação e relaxação de variáveis binárias.

2.4 Problema Discreto de Dimensionamento e Programação dos Lotes (PDDPL)

O PDDPL considera que os períodos de tempo t presentes no PDLC correspondem à macro-períodos que, por sua vez, são divididos em micro-períodos de tamanho fixo. Por exemplo, dentro de um horizonte mensal, os macro-períodos poderiam corresponder às semanas e os micro-períodos aos dias dependendo do nível de discretização desejada para o problema. As demandas e capacidades serão estabelecida dentro destes períodos menores (micro-períodos) no PDDPL. Toda a capacidade disponível no período deverá ser utilizada ou nada será produzido (hipótese "tudo ou nada"). Como os períodos têm um tamanho pequeno, a produção de um item no PDDPL poderá levar vários períodos.

Uma formulação matemática para o PDDPL apresentada por Drexl and Kimms (1997) segue abaixo:

$$\textit{Minimizar} \sum_{j,t}(s_j z_{j,t} + h_j I_{j,t}) \tag{2.14}$$

$$I_{j,t-1} + x_{j,t} - I_{j,t} = d_{j,t} \qquad \forall j,t \tag{2.15}$$

$$p_j x_{j,t} = C_t y_{j,t}; \qquad \forall j,t \tag{2.16}$$

$$\sum_j y_{j,t} \leq 1 \qquad \forall j,t \tag{2.17}$$

$$z_{j,t} \geq y_{j,t} - y_{j,t-1} \qquad \forall j,t \tag{2.18}$$

$$y_{j,t} \in \{0,1\} \qquad \forall j,t \tag{2.19}$$

$$I_{j,t}, x_{j,t}, z_{j,t} \geq 0 \qquad \forall j,t \tag{2.20}$$

A variável binária de decisão $y_{j,t}$ indica se a máquina está ajustada para o item j no período t ou não. As demais variáveis e parâmetros seguem as definições apresentadas para o PDLC. A hipótese "tudo ou nada"é estabelecida pela restrição (2.16). A restrição (2.17) garante que no máximo um item é produzido em cada período. A produção de um novo lote é ajustada na restrição (2.18).

Salomon et al. (1991) demonstram que o PDDPL é NP-difícil quando considera-se uma única máquina processando vários itens sem tempos de preparação. Os autores, entre outras demonstrações, também provam que encontrar uma solução factível considerando tempos de preparação independentes da seqüência é NP-Completo. Porém, Bruggemann and Jahnke (2000) apresentam incorreções nos resultados obtidos por Salomon. Os autores provam que o PDDPL é NP-difícil em sentido forte (*strong sense*).

Fleischmann (1990) resolve o PDDPL utilizando um procedimento do tipo *branch & bound* com relaxação lagrangiana. O método é testado em instâncias que variavam de 3, 8 e 12 produtos com 63 até 250 períodos. Soluções ótimas foram obtidas em diversas instâncias. Quando o ótimo não foi alcançado, soluções factíveis próximas ao limitante inferior foram obtidas em poucos minutos. Fleischmann (1990) soluciona o PDDPL considerando custo de preparação dependente da seqüência. O problema é formulado como um Problema do Caixeiro Viajante com Janelas de Tempo (PCVJT). Uma heurística do tipo *backward* é inicialmente utilizada para construir a solução. Em seguida, outra heurística é responsável por melhorar a solução inicial. Instâncias com 10 produtos e 150 períodos são solucionadas.

Salomon et al. (1997) resolvem o PDDPL com custo e tempo de preparação dependentes da seqüência. Os autores também formulam o problema como um PCVJT e utilizam um método de programação dinâmica proposto por Dumas et al. (1995). As instâncias avaliadas variavam desde 3 itens com 20 períodos até 10 itens com 60 períodos. O método mostrou-se sensível ao tamanho do problema. Entre outros aspectos, os autores observaram que em instâncias onde o número de períodos aumentava, o tempo computacional crescia mais rapidamente que em instâncias onde se aumentava o número de itens.

Um novo tipo de modelo pode ser estabelecido relaxando a hipótese "tudo ou nada"e permitindo que nem toda capacidade do período seja utilizada. Apenas um único item é produzido em cada período, mas não necessariamente consome toda capacidade disponível. Drexl and Kimms (1997) chamam um problema com essa nova característica de Problema Contínuo de Dimensionamento de Lote (PCDL). O termo contínuo refere-se ao fato de que os lotes agora podem assumir qualquer valor contínuo. Os lotes deixam de ser múltiplos dos valores produzidos por período. O PCDL segue a mesma formulação matemática do PDDPL, apenas a restrição (2.16) é alterada para:

$$p_j x_{j,t} \leq C_t y_{j,t}$$

Abordagens para o PCDL podem ser encontradas nos trabalhos de Bitran and Matsuo (1986) e Karmakar and Kekre (1987).

2.5 Problema Proporcional de Dimensionamento e Programação dos Lotes (PPDPL)

O PPDPL diferencia-se do PDDPL ao permitir que até dois itens sejam produzidos em um mesmo período. Um item atribuído ao período não necessariamente utiliza toda a capacidade disponível, conforme foi estabelecido no PCDL. Todavia, a capacidade remanescente poderá ser utilizada por um segundo item no PPDPL. A divisão proporcional da capacidade disponível do período entre dois itens é o que motivou o nome deste problema.

A formulação matemática sofre pequenas alterações em relação ao PDDPL. A variável $y_{j,t} = 1$ passa a indicar que o item j é produzido pela máquina no final do período t. Logo, j será o primeiro item produzido em $t + 1$. A restrição (2.16) é substituída por:

$$p_j x_{j,t} \leq C_t(y_{j,t-1} + y_{j,t}) \qquad \forall t \forall j \tag{2.21}$$

$$\sum_j p_j x_{j,t} \leq C_t \qquad \forall t \tag{2.22}$$

A restrição (2.21) estabelece que um item será produzido caso seja atribuído à máquina no início ou no fim do período t. Como mais de um item pode ser produzido em um mesmo período, a restrição (2.22) garante que a capacidade do período não seja violada.

Drexl and Haase (1995) propõem um procedimento do tipo *backward* para resolver o PPDPL. No método, chamado BACKADD, o tamanho e a programação dos lotes são determinados para $t = T, T-1, ..., 1$ e dois itens são programados em cada período t. Um valor $\rho_{j,t}$ é estabelecido para decidir quais itens serão produzidos em t. No cálculo de $\rho_{j,t}$, utilizam-se parâmetros que procuram avaliar o fator de arrependimento por não produzir o item j em t. Itens com maior valor de $\rho_{j,t}$ terão mais chances de serem produzidos. Os autores avaliam um total de 90 instâncias com $J = 2, 3, 4$ e macro-períodos $T = 3, 4, 5, 12$ divididos em $2, 3$ ou 4 micro-períodos. Esses parâmetros são combinados gerando as diversas instâncias. O método obteve soluções com um desvio médio em relação ao ótimo de 2,5%.

Drexl and Haase (1996) modificaram o método anterior que agora passa a utilizar uma análise seqüêncial na determinação dos parâmetros envolvidos no cálculo de $\rho_{j,t}$. O método foi chamado BACKADD/SEQ e testado nas mesmas instâncias que BACKADD. BACKADD na média de todos os resultados ficou 0,12% acima daqueles obtidos por BACKADD/SEQ. O número de resultados infactíveis obtidos foi reduzido de 17% em BACKADD para 13% em BACKADD/SEQ.

Kimms (1999) resolve o PPDPL multi-nível, onde a produção de um item (item final) requer certa quantidade de outros itens produzidos em estágios anteriores (itens intermediários). Há um custo de preparação que independe da seqüência e não ocorre tempo de preparação. Existem várias máquinas

e, dependendo do item, sua produção pode ocorrer em uma única máquina ou compartilhada em várias máquinas. As modificações no modelo do PPDPL estão apresentadas nas restrições abaixo:

$$I_{j,t} + x_{j,t} - I_{j,t-1} = d_{j,t} - \sum_{i \in \Im_j} a_{i,j} x_{it} \qquad \forall j, t \tag{2.23}$$

$$I_{j,t} \geq \sum_{i \in \Im_j} \sum_{\tau=t+1}^{min\{t+v_j, T\}} a_{j,i} x_{i,\tau} \qquad \forall j, t \tag{2.24}$$

$$\sum_{j \in \aleph_m} p_j x_{j,t} \leq C_{m,t} \qquad \forall m, t \tag{2.25}$$

$$\sum_{j \in \aleph_m} y_{j,t} \leq 1 \qquad \forall m, t \tag{2.26}$$

A restrição (2.23) inclui a demanda pelos itens intermediários. O fator $a_{j,i}$ determina a quantidade do item j necessária à produção do item i. O conjunto $\Im_j = \{i \in \{1, ..., J\} | a_{j,i} > 0\}$ define os itens que necessitam de certa quantidade de j em sua produção. A restrição (2.24) garante que, ao final de cada período, esteja no estoque do item j uma quantidade suficiente para produzir outros itens nos próximos v_j períodos (v_j é o *lead time* do item j). A restrição (2.25) garante que a capacidade da máquina em t não seja ultrapassada. O conjunto $\aleph_m = \{j \in \{1, ..., J\} | m_j = m\}$ determina os itens que compartilham a mesma máquina m, onde m_j é a máquina na qual o item j é produzido. A restrição (2.26) determina que ocorre no máximo um único tipo de ajuste para cada máquina ao final de cada período.

O autor utiliza Algoritmo Genético para resolver o PPDPL, onde os seguintes aspectos são destacados:

- Um cromossomo (solução) é representado por uma matriz com M linhas (número de máquinas) e T colunas (número de períodos). Uma população é formada por um conjunto de matrizes, onde cada matriz é identificada por um índice k. Uma entrada $\vartheta_{(m,t),k}$ de uma matriz k representa a regra que seleciona o tipo de estado da máquina m no fim do período t.

- A função de *fitness* utilizada é a função objetivo do problema.

- O *crossover* seleciona duas matrizes da população. Cada uma delas é cortada em quatro pedaços obtendo-se submatrizes. As submatrizes diferentes de cromossomos pais são colocadas juntas gerando cromossomos filhos.

O método é avaliado em um conjunto de 1080 instâncias com $J = 5$, $T = 10$ e $M = 1, 2$. O AG encontra soluções factíveis para um total de 1033 instâncias. O método apresentou um tempo de resolução médio de 0,10 segundos e um desvio médio do ótimo em torno de 19,98%. O autor compara

o AG com os resultados obtidos utilizando Busca Tabu (Kimms (1997)). O AG supera os resultados obtidos pela Busca Tabu tanto em tempo computacional quanto em capacidade de encontrar soluções factíveis. O desvio médio do ótimo também foi um pouco melhor do que o obtido pela Busca Tabu.

2.6 Problema Geral de Dimensionamento e Programação de Lote (PGDPL)

Este problema foi inicialmente estabelecido por Fleischmann and Meyr (1997) com as seguintes características:

- Determinar o dimensionamento e a programação de lotes de vários produtos em uma única máquina.
- O horizonte de planejamento T é dividido em macro-períodos $t = 1, 2, ..., T$. Por sua vez, cada macro-período t contém um conjunto de lotes s.
- A capacidade disponível da máquina é estabelecida por macro-período.
- As demandas são estabelecidas por macro-período e devem sempre ser atendidas.
- Os custos de preparação dependem da seqüência de produção dos itens.
- Deve-se minimizar o custo de estoque e o custo de preparação.
- Os seguintes parâmetros são considerados:
  1. S_t: conjunto lotes s pertencentes ao macro-período t.
  2. m_j: quantidade mínima de itens j a ser produzida.
  3. $sc_{i,j}$: custo de preparação do item j após o item i.
  4. $I_{j,0}$: estoque inicial do produto j.
  5. y_{j0}: igual a 1, se a máquina está ajustada inicialmente para o item j (0, caso contrário).
- As seguintes variáveis de decisão são consideradas:
  1. $x_{j,s}$: quantidade do item j no lote s.
  2. $y_{j,s}$: indica se item j está ajustado ($y_{j,s} = 1$) ou não ($y_{j,s} = 0$) no lote s da máquina.
  3. $z_{i,j,s}$: indica se ocorre ou não uma troca na produção do item i pelo item j no início do lote s.

Uma formulação matemática para o PGDPL é a seguinte:

$$\textit{Minimizar} \sum_{j,t} h_j I_{j,t} + \sum_{i,j,s} sc_{i,j} z_{i,j,s} \tag{2.27}$$

$$I_{j,t} = I_{j,t-1} + \sum_{s \in S_t} x_{j,s} - d_{j,t} \qquad \forall t, j \tag{2.28}$$

$$\sum_{j,s \in S_t} p_j x_{j,s} \leq C_t \qquad \forall t \tag{2.29}$$

$$x_{j,s} \leq \frac{C_t}{p_j} y_{j,s} \qquad \forall s, j \tag{2.30}$$

$$x_{j,s} \geq m_j (y_{j,s} - y_{j,s-1}) \qquad \forall s, j \tag{2.31}$$

$$\sum_{j} y_{j,s} = 1 \qquad \forall s \tag{2.32}$$

$$z_{i,j,s} \geq y_{i,s-1} + y_{j,s} - 1 \qquad \forall s, i, j \tag{2.33}$$

$$I_{j,t}, x_{j,s}, z_{i,j,s} \geq 0 \qquad \forall i, j, s, t \tag{2.34}$$

$$y_{j,s} \in \{0, 1\} \qquad \forall j, s \tag{2.35}$$

A função objetivo (2.27) minimiza os custo de preparação e estoque. As restrições (2.28) e (2.29) são, respectivamente, a equação de balanceamento do estoque e a restrição de capacidade das máquinas. A restrição (2.30) garante que a produção de um item em um lote ocorra apenas se a máquina estiver ajustada para produzi-lo. A restrição (2.31) impõe que a troca de itens na máquina leve pelo menos à produção da quantidade mínima do novo item. A restrição (2.32) determina que um único item seja atribuído a cada lote, ainda que nada venha a ser produzido no lote. A restrição (2.33) estabelece se ocorre ou não troca de itens no lote s.

Cada macro-período t possui no máximo $|S_t|$ lotes e esse número total de lotes estabelece um limite para a quantidade de itens programados em t. Um único item j é atribuído a cada lote s, portanto, a partir da quantidade armazenada em s temos o tempo de produção de j em t. Se $x_{j,s}$ é a quantidade do item j no lote s em t, o tempo de produção desse lote é $p_j x_{j,s}$. Assim, os macro-períodos t estão divididos em $|S_t|$ micro-períodos de diferentes tamanhos. O tamanho de cada micro-período em t é dado pelo valor de $p_j x_{j,s}$, quando $x_{j,s} > 0$.

Fleischmann and Meyr (1997) usam uma heurística do tipo *Threshold accepting* (TA) para solucionar o PGDPL. Inicialmente, as variáveis binárias $y_{j,s}$ são fixadas estabelecendo uma atribuição de itens à máquina. TA busca soluções candidatas na vizinhança desta atribuição, aplicando uma série de operações (inserção, eliminação e troca) sobre a mesma. Se a nova atribuição gerar uma solução melhor, trabalha-se com ela a partir de então. Para evitar convergência prematura, uma atribuição pior

é aceita caso a queda na qualidade da solução não ultrapasse um determinado limite. Cada vez que uma atribuição é fixada, precisa-se determinar as quantidades a serem produzidas e estocadas. Neste caso, um Problema de Fluxo em Rede (PFR) é resolvido. Os autores solucionam o PFR utilizando uma heurística do tipo *backward*.

Fleischmann avaliou três variantes do método proposto: SIM, MOD e CAP. O método MOD diferencia-se de SIM (método padrão) por estimar a capacidade disponível em períodos anteriores, durante a determinação das quantidades a serem produzidas. Tanto SIM quanto CAP aceitam soluções infactíveis que são penalizadas. CAP diferencia-se dos demais métodos por considerar apenas soluções factíveis. Os métodos são testados em um conjunto de instâncias consideradas de pequena dimensão: $J = 4$ com $T = 5, 6$ e $J = 5$ com $T = 5$. SIM e MOD apresentam resultados médios com menos de 7% de desvio da solução ótima. Considerando apenas os melhores resultados obtidos, SIM e MOD superaram aqueles conseguidos por Haase (1996) e ficam a menos de 1% do valor ótimo. Nas instâncias consideradas maiores, $J = 3$, 4 com $T = 26$ e $J = 8$, 9 com $T = 8$, os melhores resultados de SIM e MOD também superaram os obtidos em Haase (1996). CAP apresentou resultados inferiores porque, ao não aceitar soluções infactíveis, acaba convergindo para ótimos locais.

Uma extensão do PGDPL que inclui tempos de preparação dependentes da seqüência é apresentada por Meyr (2000). Dois métodos são propostos, onde o primeiro continua utilizando TA e o segundo trabalha com *Simulated Annealing*(SA). Neste caso, após serem fixadas as variáveis binárias, o PFR é resolvido na otimalidade. Utiliza-se um procedimento de reotimização baseado em um algoritmo dual proposto por Ali et al. (1989). Em linhas gerais, gera-se uma seqüência crescente de limites inferiores para o custo mínimo do PFR. Estes limites permitem que se recuse previamente algumas soluções candidatas, evitando que seja resolvido todo o PFR.

Os métodos são inicialmente testados em instâncias que não consideram tempo de preparação. As instâncias apresentam $J = 4$, 5 e $T = 5$, 6; $J = 3$, 4 e $T = 26$; $J = 9$ e $T = 8$. O autor comparou os resultados com aqueles obtidos pela heurística propostas em Haase (1993) e Fleischmann (1990). SA e TA superam os resultados daqueles trabalhos inclusive em termos de tempo computacional, mesmo com o PFR sendo resolvido na otimalidade. As instâncias envolvendo tempo de preparação apresentavam: $J = 4$ com $T = 5$ e $J = 2, 3, ..., 16$ com $T = 4$. TA e SA também alcançam os melhores resultados quando comparados a uma versão de MOD (Fleischmann (1990)) que considera tempos de preparação. Além disso, observou-se um melhor desempenho de TA em instâncias menores e SA em instâncias maiores.

Meyr (2002) estende o PGDPL com tempo de preparação para o contexto de máquinas paralelas. Os seguintes aspectos devem ser considerados na formulação matemática deste problema:

1. $S_{l,t}, C_{l,t}, p_{l,t}, m_{l,t}$, ...etc: Estes parâmetros têm o mesmo significado do modelo anterior, porém, devem ser interpretados em relação a cada linha de produção l.

2. $x_{l,j,s}, y_{l,j,s} e z_{l,i,j,s}$: as variáveis de decisão têm o mesmo significado do modelo anterior, porém, interpretadas em relação a cada linha de produção l.

A formulação matemática segue abaixo:

$$\textit{Minimizar} \sum_{j,t} h_j I_{j,t} + \sum_{l,i,j,s} s_{l,i,j} z_{l,i,j,s} \tag{2.36}$$

$$I_{j,t} = I_{j,t-1} + \sum_{s \in S_{l,t}} x_{l,j,s} - d_{j,t} \qquad \forall t, j \tag{2.37}$$

$$\sum_{j,s \in S_t} \sum_{j,s,\in S_{l,t}} p_{l,j} x_{l,j,s} \leq C_{l,t} - \sum_{i,j,s \in S_{l,t}} st_{l,i,j} z_{l,i,j,s} \qquad \forall l, t \tag{2.38}$$

$$x_{l,j,s} \leq \frac{C_{l,t}}{p_{l,j}} y_{l,j,s} \qquad \forall l, s, j \tag{2.39}$$

$$x_{l,j,s} \geq m_{l,j}(y_{l,j,s} - y_{l,j,s-1}) \qquad \forall l, s, j \tag{2.40}$$

$$\sum_{j} y_{l,j,s} = 1 \forall l, s \tag{2.41}$$

$$z_{l,i,j,s} \geq y_{l,i,s-1} + y_{l,j,s-1} - 1 \qquad \forall l, s, i, j \tag{2.42}$$

$$I_{l,j,t}, x_{l,j,s}, z_{l,i,j,s} \geq 0 \qquad \forall l, i, j, s, t \tag{2.43}$$

$$y_{l,j,s} \in \{0, 1\} \qquad \forall l, j, s \tag{2.44}$$

A função objetivo e todas as restrições apresentam a mesma interpretação do PGDPL anterior, porém considerando-se cada máquina. A restrição (2.38) inova em relação ao modelo anterior, pois retrata o impacto dos tempos de preparação na capacidade das máquinas. O método de resolução também utiliza SA e TA como heurísticas de busca local. Ao serem fixadas as variáveis binárias, além da programação da produção em cada linha, são determinadas a atribuição dos itens às máquinas. Um Problema Geral de Fluxo em Rede (Ahuja et al. (1993)) é resolvido. O método solucionou 12 instâncias de problemas práticos com $T = 8$ períodos, $L = 2$ linhas de produção e J=15 até 19 itens. Os resultados foram melhores que aqueles obtidos manualmente. TA e SA forneceram soluções com a mesma qualidade, porém TA apresenta menor tempo computacional. Também são solucionadas as intâncias CHES (Baker and Muckstadt (1989)). Os resultados obtidos foram melhores ou bastante próximos dos alcançados por Kang et al. (1999).

2.7 Conclusão

A revisão bibliográfica concentrou-se em publicações que tratavam do Problema Conjunto de Dimensionamento de Lote e Programação da Produção (PCDLPP). Problemas e modelos com aspectos semelhantes ao problema que motiva esta tese foram priorizados. Observou-se que poucos trabalhos tratam do problema de programação e dimensionamento de lotes considerando máquinas paralelas, custos e tempos de preparação dependentes da seqüência. Vários artigos tratam do caso de uma única máquina com custos e tempo de preparação dependentes da seqüência. Alguns artigos tratam do dimensionamento e programação envolvendo máquinas paralelas.

Dentre as abordagens adotadas, os modelos podem ser separados naqueles que programam vários itens por períodos e nos que programam no máximo dois itens por períodos. Trata-se de como enfocar a divisão do horizonte de planejamento. Destacamos o PGDPL que considera a atribuição dos itens a uma quantidade limitada de lotes por período. Isso leva a uma divisão do período em função da quantidade a ser produzida nos lotes, ou seja, uma divisão do período em micro-períodos de tamanho variável. Por outro lado, a divisão prévia do período em micro-períodos de tamanho fixo parece permitir um maior controle no tempo das atribuições dos itens dentro dos períodos. Essa é a abordagem apresentada no PDDPL, PCDL e PPDPL.

A revisão também mostrou que os métodos heurísticos são amplamente utilizados na resolução de instâncias consideradas de elevada dimensão. À medida que parâmetros como J e T crescem, aumentando a dimensão das instâncias e a complexidade dos modelos relacionados, abordagens que utilizam métodos exatos começam a encontrar dificuldade em determinar soluções dentro de um tempo computacional razoável.

A presente revisão permitiu que se adquiresse maior visão desta área de estudo e lançou luz sobre abordagens mais adequadas a serem utilizadas no decorrer deste trabalho.

Capítulo 3

Modelo Matemático

3.1 Introdução

O problema industrial descrito no capítulo 1 apresenta diversas restrições a serem satisfeitas e alguns objetivos a serem atingidos. O presente capítulo irá sumarizar as diversas restrições e objetivos envolvidos através de uma formulação matemática do problema. Um modelo matemático como o proposto nesse capítulo atende a diversos propósitos. Primeiro, ele procura descrever matematicamente os objetivos e restrições envolvidas no problema estudado. Segundo, uma vez estabelecido matematicamente, o modelo pode ser codificado em uma ferramenta computacional de modelagem matemática. Isso permitirá encontrar soluções ótimas para instâncias de pequena dimensão, dado que instâncias de elevada dimensão são difíceis de serem resolvidas por *solvers* dos pacotes comerciais. Essas soluções serão usadas como parâmetro na avaliação de soluções obtidas por métodos heurísticos. Por último, o modelo proposto pode ser usado em futuras pesquisas envolvendo abordagens baseadas em programação matemática.

O modelo matemático proposto inova ao utilizar simultaneamente aspectos presentes no Problema Geral de Dimensionamento e Programação de Lote (PGDPL) e no Problema Contínuo de Dimensionamento de Lote (PCDL) na sua formulação. Um horizonte de tempo T foi dividido em macro-períodos de tamanho fixo t como ocorre tanto no PGDPL como no PCDL. As demandas existentes para os produtos são atendidas ao final de um macro-período. Logo, a divisão do horizonte T em macro-períodos $t = 1, ..., T$ permite localizar no tempo quando as demandas devem ser satisfeitas.

Um número fixo de lotes (S para linhas e $\overline{S}$ para tanques) foi considerado em cada macro-período, utilizando assim a idéia presente no PGDPL (Fleischmann (1990)) de se programar um número máximo de lotes por macro-período tanto nas linhas ($|S|$) como nos tanques ($|\overline{S}|$). Isso permitirá controlar a atribuição dos produtos e xaropes às linhas e tanques dentro de cada macro-período, bem como a seqüência de produção e armazenagem através da ordem de ocupação dos lotes das linhas e tanques

disponíveis dentro dos macro-períodos.

A figura 3.1 exemplifica essa idéia. Para isso, consideramos dois macro-períodos com no máximo $S = \overline{S} = 2$ lotes. Um total de 5 xaropes e 6 produtos foram dimensionados e programados em 3 tanques e 3 linhas, respectivamente. O xarope XpA produz o produto P1, XpB produz P2 e P3, XpC produz P4, XpD produz P5 e XpE produz P6. Observamos no exemplo que as atribuições de produtos

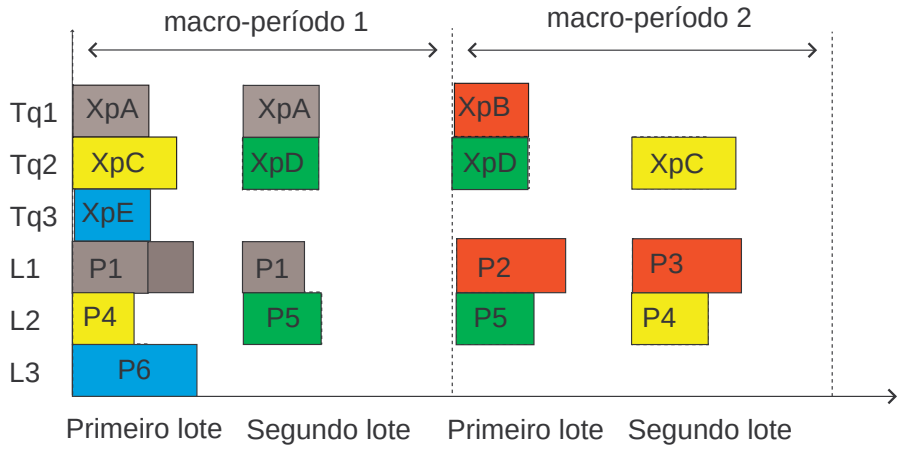

Fig. 3.1: Ocupação dos lotes nos tanques e linhas.

às linhas e de xaropes aos tanques não ultrapassam o número máximo possível ($S = \overline{S} = 2$). Nem todos os lotes disponíveis foram ocupados na figura 3.1. Por exemplo, somente um lote é ocupado por xarope e produto, respectivamente, no tanque Tq3 e na linha L3 durante o primeiro macro-período. No segundo macro-período, nenhum lote é ocupado em Tq3 e L3. As diferenças de tamanho nos lotes apenas ilustram que as quantidades de produtos atribuídas a cada lote podem ser diferentes.

A existência de dois níveis interdependentes exige que seja estabelecida uma sincronia entre o que é programado nas linhas com o que é programado nos tanques. Essa sincronia é necessária já que, por exemplo, um produto não pode ser produzido nas linhas sem que o respectivo xarope esteja programado no tanque. Para isso, utilizamos a idéia presente no PDDPL, PCDL e PPDPL de se dividir os macro-períodos em micro-períodos. Cada macro-período é discretizado em um número igual de micro-períodos de mesma capacidade. Isso permitirá um controle de quando cada lote está sendo produzido e armazenado, respectivamente, nas linhas e tanques. Assim, variáveis e restrições que permitem sincronizar a produção das linhas com o armazenamento de xaropes nos tanques podem ser estabelecidas.

A fim de facilitar a formulação matemática, estabelecemos que cada micro-período seja ocupado por no máximo um produto que não precisará utilizar toda capacidade do micro-período, seguindo dessa forma a hipótese do PCDL. Também iremos estabelecer que um lote programado em um tanque esteja disponível pelo menos um micro-período antes do início da sua produção nas linhas. Isso

evita que um produto seja produzido sem que o respectivo xarope esteja previamente armazenado no tanque.

A figura3.2 ilustra, por meio de um gráfico de Gantt, o uso dos micro-períodos na sincronia dos lotes apresentados anteriormente na figura 3.1. Cada macro-período foi dividido em 5 micro-períodos

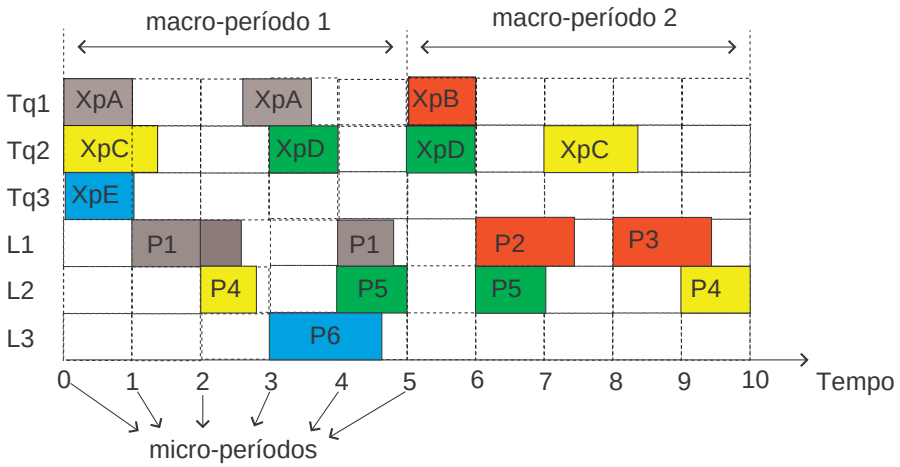

Fig. 3.2: Sincronia de lotes entre tanques e linhas.

de mesmo tamanho. XpA, XpB, XpC, XpD e XpE na figura 3.2 indicam o tempo de preparação do xarope no tanque e P1, P2, P3, P4 e P5 indicam o tempo de processamento dos produtos. Os dois lotes do tanque Tq1 são ocupados pelo xarope XpA, ou seja, o tanque será abastecido e reabastecido pelo mesmo xarope durante o primeiro macro-período. Essa ocupação de Tq1 visa atender a produção de P1 na linha L1, onde os dois lotes disponíveis no primeiro macro-período são ocupados pelo produto P1. Os micro-períodos permitem sincronizar o início da produção de P1 com o final do preparo do primeiro lote de XpA, assim como também permitem localizar quando ocorrerá o reabastecimento de Tq1 com XpA (segundo lote). Esse reabastecimento exige a interrupção de P1 e o início do segundo lote de P1 após o término do preparo de XpA em Tq1.

Todos os produtos começam a ser produzidos nas linhas nos micro-períodos seguintes àqueles em que os xaropes estão prontos nos tanques. Todavia, conforme ilustrado no tanque Tq3 e linha L3, a produção não precisa começar imediatamente após o preparo do tanque. Na linha L1, destacamos que no segundo macro-período a produção de P3 começa no micro-período seguinte ao término de P2 já que estamos supondo não haver produção de dois itens em um mesmo micro-período.

Resumidamente, podemos afirmar que a inovação proposta pelo modelo apresentado neste capítulo está no uso simultâneo e sincronizado de lotes e micro-períodos na sua formulação. Os lotes determinam a seqüência de ocupação dos tanques dentro dos macro-períodos. Os micro-períodos estabelecem no tempo(e de forma sincronizada) a ocupação dos lotes nas linhas e nos tanques dentro de cada

macro-período. Ressaltamos que abordagem similar não foi encontrada na revisão bibliográfica apresentada no capítulo 2, por isso, o modelo aqui proposto inova em termos de formulação matemática para um problema com dois níveis interdependentes.

Pelo exemplo anterior, podemos perceber que programar e dimensionar os lotes de produtos e xaropes de forma sincronizada não é uma tarefa fácil. A formulação matemática faz uso de diversas variáveis, principalmente variáveis binárias, indexadas nos micro-períodos e nos lotes para representar todas as restrições envolvidas no problema aqui estudado. A notação das variáveis e parâmetros envolvidos é apresentada na próxima seção.

3.2 Notação Utilizada

O significado de cada parâmetro e variável de decisão presente no modelo é apresentado nesta seção.

3.2.1 Parâmetros Gerais

T Número de macro-períodos.

C Capacidade (em unidades de tempo) dentro de um macro-período.

T^m Número de micro-períodos por macro-períodos.

C^m Capacidade (em unidades de tempo) dentro de um micro-período ($C = T^m C^m$).

ϵ Um número real positivo muito pequeno.

M Um número real positivo muito grande.

3.2.2 Parâmetros para as Linhas de Produção

J Número de produtos, onde $i, j = 1, ..., J$.

L Número de linhas em paralelo, onde $l = 1, ..., L$.

L_j Conjunto de linhas nas quais o produto j pode ser produzido.

S Número máximo de lotes a ser ocupado por produtos dentro de cada macro-período, onde $s = 1, ..., S$.

d_{jt} Demanda pelo produto j no final do macro-período t.

s_{ijl} Custo de troca dependente da seqüência de produção do produto i para o produto j na linha l.

h_j Custo de estoque para cada unidade do produto j que permanece em estoque ao final de um macro-período.

v_{jl} Custo de produção de uma unidade do produto j na linha l.

p_{jl} Tempo de processamento de uma unidade do produto j na linha l (hipótese: $p_{jl} \leq C^m$).

x_{jl0} = 1, Se linha l é ajustada para o produto j no início do horizonte de produção; 0, caso contrário.

σ'_{ijl} Tempo de troca (*setup time*) para ajustar linha l a partir do produto i para o produto j (hipóteses: $\sigma'_{ijl} \leq C$ e $\sigma'_{jjl} = 0$).

ω_{l1} Tempo de troca para o primeiro lote na linha l no macro-periodo 1 que já foi executado antes do macro-período 1 começar (Observação: se $\omega_{l1} > 0$, então os valores x_{jl1} são conhecidos e estas variáveis devem ser ajustadas adequadamente).

I_{j0} Estoque inicial do produto j.

3.2.3 Variáveis de decisão para as Linhas de Produção

z_{ijls} Indica se a linha l é ajustada ($z_{ijls} = 1$) ou não ($z_{ijls} = 0$) a partir do produto i para o produto j no lote s (definir $z_{ijls} \geq 0$ é suficiente).

I_{jt} Quantidade de produto j em estoque no final do macro-periodo t.

q_{jls} Número de produtos j produzidos na linha l no lote s.

x_{jls} = 1, se o lote s na linha l pode ser usado para produzir o produto j; 0, caso contrário.

u_{ls} = 1, se uma quantidade é efetivamente produzida no lote s na linha l; 0, caso contrário.

σ_{ls} Tempo de troca na linha l que ocorre antes do produção do lote s.

ω_{lt} Parte do tempo de troca que antecede a produção do primeiro lote na linha l em t que ocorre ao final do macro-período $t-1$.

$x^E_{ls\tau}$ = 1, se o lote s na linha l termina no micro-período τ.

$x^B_{ls\tau}$ = 1, se o lote s na linha l começa no macro-período τ.

δ_{ls} Tempo no primeiro micro-período do lote que é reservado para tempo de troca e tempo ocioso.

q_j^0 número de produtos j que deixaram de ser produzidos, ou seja, será positivo toda vez que a demanda do produto não for satisfeita.

3.2.4 Parâmetros para os Tanques

$\bar{J}$ Número de xaropes, onde $i, j = 1, ..., \bar{J}$.

$\bar{L}$ Número de tanques, onde $k = 1, ..., \bar{L}$.

$\bar{L}_j$ Conjunto de tanques nos quais o xarope j pode ser armazenado.

$\bar{S}$ Número máximo de lotes a ser ocupado por xaropes dentro de cada macro-período, onde $s = 1, ..., \bar{S}$.

$\bar{s}_{ijk}$ Custo de troca dependente da seqüência para uma troca partindo do xarope i para o xarope j no tanque k.

$\bar{h}_j$ Custo de estoque para cada unidade do xarope j que permaneça no tanque ao final de um macro-período.

$\bar{v}_{jk}$ Custo de produção para cada unidade do xarope j utilizada no tanque k.

$\bar{x}_{jk0}$ = 1, se o tanque k é ajustado para o xarope j no início do horizonte de planejamento; 0, caso contrário.

$\bar{\sigma}'_{ijk}$ Tempo de troca do xarope i para o xarope j no tanque k (hipóteses: $\bar{\sigma}'_{ijk} \leq C$ e $\bar{\sigma}'_{ijk}$ é um inteiro múltiplo de C^m).

$\bar{\omega}_{k1}$ Tempo de troca para o primeiro lote no tanque k no macro-período 1 realizado antes do macro-período 1 começar (Obs: se $\bar{\omega}_{k1} > 0$, então os valores $\bar{x}_{jk1}$ já são conhecidos e essas variáveis devem ser ajustadas adequadamente).

$\bar{I}_{jk0}$ Estoque inicial do xarope j no tanque k.

$\overline{Q}_k$ Quantidade máxima que poderá ser armazenada no tanque k.

$\underline{Q}_k$ Quantidade mínima que deve ser armazenada no tanque k.

3.2.5 Variáveis de decisão para os Tanques

$\bar{z}_{ijks}$ Indica se o tanque k é ajustado ($\bar{z}_{ijks} = 1$) ou não ($\bar{z}_{ijks} = 0$) a partir do xarope i para o xarope j no início do lote s. Definir $\bar{z}_{ijks} \geq 0$ é suficiente de acordo com as equações (3.1) e (3.5) do modelo matemático.

$\bar{I}_{jk\tau}$ Quantidade de xarope j no tanque k ao final do micro-período τ.

$\bar{q}_{jks}$ Quantidade de xarope j armazenada no tanque k no lote s.

$\bar{q}_{jks\tau}$ Quantidade de xarope j armazenada no tanque k no lote s no micro-período τ.

$\bar{x}_{jks}$ = 1, se o lote s pode ser usado para armazenar o xarope j no tanque k; 0, caso contrário.

$\bar{u}_{ks}$ = 1, se lote s é usado efetivamente para armazenar xarope no tanque k; 0, caso contrário.

$\bar{\sigma}_{ks}$ Tempo de troca gasto no tanque k no início do lote s.

$\bar{\omega}_{kt}$ Tempo de troca gasto antes do primeiro lote no tanque k no macro-periodo t que é programado ao final do macro-período $t-1$.

$\bar{x}^{E}_{ks\tau}$ = 1, se o tempo de troca para o xarope no lote s do tanque k termina no micro-período τ.

$\bar{x}^{B}_{ks\tau}$ = 1, se o tempo de troca para o xarope no lote s do tanque k começa no micro-período τ.

3.2.6 Parâmetros para ajustar Linhas de Produção e Tanques

r_{ji} Fator de conversão, ou seja, quantidade de xarope j necessária para produzir uma unidade do produto i.

3.2.7 Variáveis de decisão para ajustar Linhas de Produção e Tanques

$q_{kjls\tau}$ Quantidade de produto j, produzido na linha l, no micro-período τ e que pertence ao lote s, que utiliza xarope a partir do tanque k.

3.3 Formulação Matemática

A formulação matemática começa a descrição do problema pela sua função objetivo e pelas restrições relativas às linhas de produção. A seguir, as restrições relativas aos tanques são apresentadas. As restrições que descrevem a interdependência entre os dois níveis são apresentadas ora entre as restrições das linhas de produção e ora entre as restrições dos tanques.

Um total de 65 equações foram estabelecidas para descrever o Problema Conjunto de Dimensionamento de Lotes e Programação da Produção (PCDLPP) proposto. As 29 primeiras restrições se concentram nas linhas e as 36 restantes nos tanques.

A tabela 3.1 agrupa as restrições de acordo com seu significado tanto nas linhas quanto nos tanques. O objetivo é fornecer uma visão geral das restrições comuns aos dois níveis, das restrições específicas a cada nível e das restrições que estabelecem a sincronia entre linhas e tanques.

Linhas	Tanques	Significado das Restrições
3.2	3.30	Atribuições não permitidas a linhas ou tanques
3.3	3.31	Atribuições de produtos ou xaropes aos lotes
3.4,3.16 e 3.17	3.32 e 3.33	Ligações entre atribuição de produtos ou xaropes aos lotes e as quantidades produzidas ou armazenadas
3.5	3.35 e 3.36	Trocas entre produtos ou xaropes
3.6-3.8	3.37 e 3.38	Tempo de troca
3.9	3.39	Uso da capacidade do macro-período
3.10 e 3.11	3.41 e 3.42	Balanceamento de estoque
3.12 e 3.13	3.43-3.45	Uso da capacidade disponível entre o fim de dois lotes sucessivos
3.14 e 3.15	3.46, 3.48-3.50	Estabelece um único início e único fim para cada lote
3.18 - 3.21, 3.23, 3.24	3.51 - 3.59, 3.61-3.63	Estabelece o início e fim de um lote efetivamente usado ou não
3.22	3.60	Ordem de ocupação dos lotes
3.25-3.27	3.40-3.42, 3.47	Sincroniza a retirada de xarope dos lotes com a produção dos produtos nos lotes das linhas
3.28 e 3.29	-	Ocupação da capacidade do primeiro micro-período de cada lote
-	3.42, 3.64 e 3.65	Reabastecimento dos tanques

Tab. 3.1: Agrupamento das restrições pelo seu significado

A formulação matemática a seguir procura minimizar os custos de produção, preparo e estoque envolvidos. A primeira linha da função objetivo apresenta os custos relativo às linhas de produção e a segunda linha apresenta os custos relativo aos tanques. Para garantir que uma solução venha a ser encontrada, as demandas que não forem atendidas serão armazenadas na variável q_j^0. As demandas não atendidas são altamente penalizadas para se evitar tal tipo de solução.

$$\sum_{i=1}^{J}\sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{s=1}^{T\cdot S} s_{ijl}z_{ijls} + \sum_{j=1}^{J}\sum_{t=1}^{T} h_j I_{jt} + \sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{s=1}^{T\cdot S} v_{jl}q_{jls}$$

$$+\sum_{i=1}^{J}\sum_{j=1}^{J}\sum_{k=1}^{\bar{L}}\sum_{s=1}^{T\cdot\bar{S}}\bar{s}_{ijk}\bar{z}_{ijks}+\sum_{j=1}^{\bar{J}}\sum_{k=1}^{\bar{L}}\sum_{t=1}^{T}\bar{h}_j\bar{I}_{jk,t\cdot T^m}+\sum_{j=1}^{\bar{J}}\sum_{k=1}^{\bar{L}}\sum_{s=1}^{T\cdot\bar{S}}\bar{v}_{jk}\bar{q}_{jks}$$
$$+M\sum_{j=1}^{J}q_j^0 \tag{3.1}$$

Nem todos os produtos podem ser produzidos em todas as linhas. Se x_{jls} é uma variável binária que indica para qual produto j um certo lote s em determinana linha l está reservado, podemos expressar que isto nunca ocorrerá da seguinte forma:

$$x_{jls}=0 \qquad \begin{array}{l} j=1,\ldots,J \\ l\in\{1,\ldots,L\}\backslash L_j \\ s=1,\ldots,T\cdot S \end{array} \tag{3.2}$$

A abordagem aqui adotada, baseada no PGDPL, exige que cada lote em um macro período seja atribuído (reservado) a um produto, mesmo que nada venha a ser produzido:

$$\sum_{j=1}^{J}x_{jls}=1 \qquad \begin{array}{l} l=1,\ldots,L \\ s=1,\ldots,T\cdot S \end{array} \tag{3.3}$$

Uma quantidade q_{jls} será programada para produção em determinada linha apenas se o lote s estiver reservado ao produto j ($x_{jls}=1$). Isto ocorrendo, não poderá exceder a capacidade C do macro-período. Essa ligação entre as variáveis x_{jls} e q_{jls} está expressa a seguir:

$$p_{jl}q_{jls}\leq Cx_{jls} \qquad \begin{array}{l} j=1,\ldots,J \\ l=1,\ldots,L \\ s=1,\ldots,T\cdot S \end{array} \tag{3.4}$$

A troca de um produto i para j numa mesma linha deve ser identificada no modelo. Assim, podemos determinar o tempo de troca σ_{ls} que ocorre antes de determinado lote s começar a ser produzido em l:

$$z_{ijls}\geq x_{jls}+x_{il,s-1}-1 \qquad \begin{array}{l} i,j=1,\ldots,J \\ l=1,\ldots,L \\ s=1,\ldots,T\cdot S \end{array} \tag{3.5}$$

$$\sigma_{ls}=\sum_{i=1}^{J}\sum_{j=1}^{J}\sigma'_{ijl}z_{ijls} \qquad \begin{array}{l} l=1,\ldots,L \\ s=1,\ldots,T\cdot S \end{array} \tag{3.6}$$

Para o primeiro lote em um macro-período t, parte do seu tempo de troca poderá ocorrer no final

do macro-período anterior ($w_{lt} \neq 0$). Se isto acontecer, o final do último lote em $t-1$ ($x^E_{l,t.S,\tau} = 1$) deverá reservar tempo suficiente para w_{lt}

$$\omega_{lt} \leq \sigma_{l,(t-1)S+1} \qquad \begin{matrix} l = 1,\ldots,L \\ t = 1,\ldots,T \end{matrix} \tag{3.7}$$

$$t \cdot C - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau x^E_{l,t\cdot S,\tau} \geq \omega_{l,t+1} \qquad \begin{matrix} l = 1,\ldots,L \\ t = 1,\ldots,T-1 \end{matrix} \tag{3.8}$$

O total do tempo gasto em cada linha não poderá exceder a capacidade C do macro-período:

$$\sum_{s=(t-1)S+1}^{t\cdot S} (\sigma_{ls} + \sum_{j=1}^{J} p_{jl} q_{jls}) \leq C + \omega_{lt} \qquad \begin{matrix} l = 1,\ldots,L \\ t = 1,\ldots,T \end{matrix} \tag{3.9}$$

As equações de balanceamento de estoque abaixo garantem que o número de produtos em estoque é igual ao que estava em estoque no início do período, mais o que foi produzido, menos a demanda do período. Como desejamos garantir a obtenção de uma solução, permitimos que q^0_j unidades sejam "produzidas"no primeiro macro-período sem o uso de capacidade.

$$I_{j1} = I_{j0} + q^0_j + \sum_{l=1}^{L}\sum_{s=1}^{S} q_{jls} - d_{j1} \qquad j = 1,\ldots,J \tag{3.10}$$

$$I_{jt} = I_{j,t-1} + \sum_{l=1}^{L} \sum_{s=(t-1)S+1}^{t\cdot S} q_{jls} - d_{jt} \qquad \begin{matrix} j = 1,\ldots,J \\ t = 2,\ldots,T \end{matrix} \tag{3.11}$$

O tempo entre o final de um lote s e o final do lote anterior $s-1$ deve ser suficiente para que ocorra a produção do produto em s mais o tempo de troca que antecede esse lote.

$$\begin{aligned} &\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau x^E_{ls\tau} \\ &- \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau x^E_{l,s-1,\tau} \geq \sigma_{ls} + \sum_{j=1}^{J} p_{jl} q_{jls} \end{aligned} \qquad \begin{matrix} l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+2,\ldots,t\cdot S \end{matrix} \tag{3.12}$$

$$\begin{aligned} &\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau x^E_{l,(t-1)S+1,\tau} \\ &\geq (t-1)C + \sigma_{l,(t-1)S+1} - \omega_{lt} + \sum_{j=1}^{J} p_{jl} q_{jl,(t-1)S+1} \end{aligned} \qquad \begin{matrix} l = 1,\ldots,L \\ t = 1,\ldots,T \end{matrix} \tag{3.13}$$

A fim de obter uma programação bem estabelecida, todo lote deverá ter um único início e um

único fim em cada linha.

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} x^E_{ls\tau} = 1 \qquad \begin{array}{l} l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+1,\ldots,t\cdot S \end{array} \tag{3.14}$$

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} x^B_{ls\tau} = 1 \qquad \begin{array}{l} l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+1,\ldots,t\cdot S \end{array} \tag{3.15}$$

Ao passo que a variável x_{jls} indica se o lote s está reservado para produzir o produto j ou não, a variável u_{ls} indica se s é de fato produzido na linha l ou não. Se um lote não é usado, então a quantidade atribuida a ele precisa ser zero. Caso contrário, uma quantidade mínima ϵ deve ser produzida:

$$\sum_{j=1}^{J} p_{jl} q_{jls} \leq C u_{ls} \qquad \begin{array}{l} l = 1,\ldots,L \\ s = 1,\ldots,T\cdot S \end{array} \tag{3.16}$$

$$\epsilon u_{ls} \leq \sum_{j=1}^{J} q_{jls} \qquad \begin{array}{l} l = 1,\ldots,L \\ s = 1,\ldots,T\cdot S \end{array} \tag{3.17}$$

Se um lote s não é usado efetivamente para produzir algum produto, seu início ($x^B_{ls\tau}$) e seu final ($x^E_{ls\tau}$) devem ser o mesmo. Logo, nenhum tempo será reservado para produção do lote:

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau x^E_{ls\tau} - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau x^B_{ls\tau} \leq T^m u_{ls} \qquad \begin{array}{l} l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+1,\ldots,t\cdot S \end{array} \tag{3.18}$$

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau x^B_{ls\tau} - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau x^E_{ls\tau} \leq u_{ls} \qquad \begin{array}{l} l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+1,\ldots,t\cdot S \end{array} \tag{3.19}$$

Para evitar redundâncias no modelo, se um lote s não é usado, ele deverá começar à direita de quando o lote $s-1$ termina:

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau x^B_{ls\tau} - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau x^E_{l,s-1,\tau} \leq T^m u_{ls} \qquad \begin{array}{l} l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+2,\ldots,t\cdot S \end{array} \tag{3.20}$$

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau x^B_{l,(t-1)S+1,\tau} - ((t-1)T^m+1)$$

$$\leq T^m u_{l,(t-1)S+1} \qquad \begin{matrix} l = 1,\ldots,L \\ t = 1,\ldots,T \end{matrix} \tag{3.21}$$

Redundâncias também são evitadas ao se obrigar que um lote $s+1$ seja usado apenas quando os lotes anteriores $s, s-1, ..., 1$ de um mesmo macro-período já tenham sido usados. Se ocorrerem lotes vazios (não usados), eles serão os últimos lotes do macro-período:

$$u_{ls} \geq u_{l,s+1} \qquad \begin{matrix} l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+1,\ldots,t\cdot S-1 \end{matrix} \tag{3.22}$$

O início e o fim de um lote devem ser escolhidos de tal forma que o início indique o primeiro micro-período no qual o lote é produzido, se há efetivamente uma quantidade atribuída a tal lote. Da mesma forma, o fim de um lote deverá indicar o último micro-período de produção do lote:

$$\epsilon u_{ls} + \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau x^E_{ls\tau} - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau x^B_{ls\tau} \leq \sum_{j=1}^{J} p_{jl} q_{jls} \qquad \begin{matrix} l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+1,\ldots,t\cdot S \end{matrix} \tag{3.23}$$

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau x^E_{ls\tau} - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau x^B_{ls\tau} \geq \sum_{j=1}^{J} p_{jl} q_{jls} - C^m \qquad \begin{matrix} l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+1,\ldots,t\cdot S \end{matrix} \tag{3.24}$$

Uma relação entre as linhas de produção e os tanques deve ser estabelecida. Inicialmente, estabelecemos qual quantidade de um produto j produzido na linha l e atribuído ao lote s vem de quais tanques k :

$$q_{jls} = \sum_{k=1}^{\bar{L}} \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} q_{kjls\tau} \qquad \begin{matrix} j = 1,\ldots,J \\ l = 1,\ldots,L \\ t = 1,\ldots,T \\ s = (t-1)S+1,\ldots,t\cdot S \end{matrix} \tag{3.25}$$

A retirada de quantidades de xaropes dos tanques para as linhas deverá ocorrer dentro dos inter-

valos de produção dos lotes nas linhas:

$$\sum_{k=1}^{\bar{L}}\sum_{j=1}^{J} p_{jl}q_{kjls\tau} \leq \sum_{\tau'=\tau}^{t\cdot T^m} C^m x^E_{ls\tau'} \qquad \begin{array}{l} l=1,\ldots,L \\ t=1,\ldots,T \\ s=(t-1)S+1,\ldots,t\cdot S \\ \tau=(t-1)T^m+1,\ldots,t\cdot T^m \end{array} \tag{3.26}$$

$$\sum_{k=1}^{\bar{L}}\sum_{j=1}^{J} p_{jl}q_{kjls\tau} \leq \sum_{\tau'=(t-1)T^m+1}^{\tau} C^m x^B_{ls\tau'} \qquad \begin{array}{l} l=1,\ldots,L \\ t=1,\ldots,T \\ s=(t-1)S+1,\ldots,t\cdot S \\ \tau=(t-1)T^m+1,\ldots,t\cdot T^m \end{array} \tag{3.27}$$

No primeiro micro-período de produção de um lote s, algum tempo δ_{ls} é reservado para troca de produtos ou trata-se de tempo ocioso. O tempo restante neste micro-período é usado para produção. Isto é levado em consideração nas seguintes restrições:

$$\begin{aligned} \delta_{ls} = &\left(\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau x^E_{ls\tau} - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau x^B_{ls\tau} + 1\right) C^m \\ &- \sum_{j=1}^{J} p_{jl}q_{jls} \end{aligned} \qquad \begin{array}{l} l=1,\ldots,L \\ t=1,\ldots,T \\ s=(t-1)S+1,\ldots,t\cdot S \end{array} \tag{3.28}$$

$$\sum_{k=1}^{\bar{L}}\sum_{j=1}^{J} p_{jl}q_{kjls\tau} \leq C^m - \delta_{ls} + (1-x^B_{ls\tau})C^m \qquad \begin{array}{l} l=1,\ldots,L \\ t=1,\ldots,T \\ s=(t-1)S+1,\ldots,t\cdot S \\ \tau=(t-1)T^m+1,\ldots,t\cdot T^m \end{array} \tag{3.29}$$

As restrições relativas aos tanques serão consideradas agora. Nem todos os xaropes dos produtos podem ser armazenados em todos os tanques. Primeiro, estabelecemos quais tanques não podem armazenar determinados xaropes:

$$\bar{x}_{jks} = 0 \qquad \begin{array}{l} j=1,\ldots,\bar{J} \\ l\in\{1,\ldots,\bar{L}\}\backslash\bar{L}_j \\ s=1,\ldots,T\cdot\bar{S} \end{array} \tag{3.30}$$

No máximo um tipo de xarope j poderá ser armazenado em cada tanque k e reservado a determi-

nado lote s:

$$\sum_{j=1}^{\bar{J}} \bar{x}_{jks} = 1 \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ s = 1, \ldots, T \cdot \bar{S} \end{array} \tag{3.31}$$

A quantidade que pode ser armazenada em um lote tem como limitante superior a capacidade máxima do tanque. Além disso, se algum lote é efetivamente armazenado no tanque ele deverá atender a uma ocupação mínima do tanque:

$$\bar{q}_{jks} \leq \overline{Q}_k \bar{x}_{jks} \qquad \begin{array}{l} j = 1, \ldots, \bar{J} \\ k = 1, \ldots, \bar{L} \\ s = 1, \ldots, T \cdot \bar{S} \end{array} \tag{3.32}$$

$$\bar{q}_{jks} \leq \overline{Q}_k \bar{u}_{ks} \qquad \begin{array}{l} j = 1, \ldots, \bar{J} \\ k = 1, \ldots, \bar{L} \\ s = 1, \ldots, T \cdot \bar{S} \end{array} \tag{3.33}$$

$$\sum_{j=1}^{\bar{J}} \bar{q}_{jks} \geq \underline{Q}_k \bar{u}_{ks} \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ s = 1, \ldots, T \cdot \bar{S} \end{array} \tag{3.34}$$

O custo e o tempo de troca envolvendo xaropes nos tanques também dependem da seqüência em que os xaropes são armazenados. Nas linhas de produção, o estado das linhas é preservado para um mesmo tipo de produto. Isto não ocorre nos tanques. O conteúdo de um tanque deve ser completamente escoado para que o mesmo seja limpo e reabastecido. Esse procedimento é seguido ainda que a troca envolva o mesmo tipo de xarope. Desta forma, tempos e custos de troca ocorrem nos tanques apenas quando o tanque é efetivamente utilizado ($\bar{x}_{jks} = 1$,$\bar{x}_{ik,s-1} = 1$,$\bar{u}_{ks} = 1$ em 3.35 abaixo). Mudanças na atribuição de um lote reservado ao xarope em um tanque ($\bar{x}_{jks} = 1$,$\bar{x}_{ik,s-1} = 1$,$\bar{u}_{ks} = 0$) não necessariamente significa que houve troca:

$$\bar{z}_{ijks} \geq \bar{x}_{jks} + \bar{x}_{ik,s-1} - 2 + \bar{u}_{ks} \qquad \begin{array}{l} i, j = 1, \ldots, \bar{J} \\ k = 1, \ldots, \bar{L} \\ s = 1, \ldots, T \cdot \bar{S} \end{array} \tag{3.35}$$

$$\bar{x}_{jks} - \bar{x}_{jk,s-1} \leq u_{ks} \qquad \begin{array}{l} j = 1, \ldots, \bar{J} \\ k = 1, \ldots, \bar{L} \\ s = 1, \ldots, T \cdot \bar{S} \end{array} \tag{3.36}$$

$$\bar{\sigma}_{ks} = \sum_{i=1}^{\bar{J}} \sum_{j=1}^{\bar{J}} \bar{\sigma}'_{ijk} \bar{z}_{ijks} \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ s = 1, \ldots, T \cdot \bar{S} \end{array} \tag{3.37}$$

$$\bar{\omega}_{kt} \leq \bar{\sigma}_{k,(t-1)S+1} \qquad t = 1, \ldots, T \tag{3.38}$$

$$\sum_{s=(t-1)\bar{S}+1}^{t\cdot\bar{S}} \bar{\sigma}_{ks} \leq C + \bar{\omega}_{kt} \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ t = 1, \ldots, T \end{array} \tag{3.39}$$

A quantidade armazenada em um tanque precisa ser programada tal que o micro-período em que o xarope será retirado do tanque seja conhecido:

$$\bar{q}_{jks} = \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \bar{q}_{jks\tau} \qquad \begin{array}{l} j = 1, \ldots, \bar{J} \\ k = 1, \ldots, \bar{L} \\ t = 1, \ldots, T \\ s = (t-1)\bar{S}+1, \ldots, t\cdot\bar{S} \end{array} \tag{3.40}$$

A quantidade em um tanque ao final de um micro-período é o que já estava no tanque, mais o que foi adicionado, menos o que escoa para as linhas. A seguir, veremos que os tanques podem ser reabastecidos apenas se estiverem vazios. Para coordenar linhas e tanques, exige-se que a quantidade retirada dos tanques esteja armazenada pelo menos um micro-período antes:

$$\begin{aligned} \bar{I}_{jk\tau} = \bar{I}_{jk,\tau-1} + \sum_{s=(t-1)\bar{S}+1}^{t\cdot\bar{S}} \bar{q}_{jks\tau} \\ - \sum_{i=1}^{J}\sum_{l=1}^{L}\sum_{s=(t-1)S+1}^{t\cdot S} r_{ji} q_{kils\tau} \end{aligned} \qquad \begin{array}{l} j = 1, \ldots, \bar{J} \\ k = 1, \ldots, \bar{L} \\ t = 1, \ldots, T \\ \tau = (t-1)T^m+1, \ldots, t\cdot T^m \end{array} \tag{3.41}$$

$$\bar{I}_{jk\tau} \geq \sum_{i=1}^{J}\sum_{l=1}^{L}\sum_{s=(t-1)S+1}^{t\cdot S} r_{ji} q_{kils,\tau+1} \qquad \begin{array}{l} j = 1, \ldots, \bar{J} \\ k = 1, \ldots, \bar{L} \\ t = 1, \ldots, T \\ \tau = (t-1)T^m, \ldots, t\cdot T^m - 1 \end{array} \tag{3.42}$$

O lote s de um tanque precisa ser programado tal que seu tempo de troca ocorra entre o final do lote $s-1$ e o final de s. Parte do tempo de troca também pode ocorrer no macro-período anterior da mesma forma que aconteceu nas linhas:

$$\begin{aligned} t\cdot C - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau \bar{x}^E_{k,t\cdot\bar{S},\tau} \geq \bar{\omega}_{k,t+1} \\ \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m \tau \bar{x}^E_{ks\tau} \end{aligned} \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ t = 1, \ldots, T-1 \end{array} \tag{3.43}$$

$$-\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m\tau\bar{x}^E_{k,s-1,\tau} \geq \bar{\sigma}_{ks} \qquad \begin{array}{l} k=1,\ldots,\bar{L} \\ t=1,\ldots,T \\ s=(t-1)\bar{S}+2,\ldots,t\cdot\bar{S} \end{array} \tag{3.44}$$

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} C^m\tau\bar{x}^E_{k,(t-1)\bar{S}+1,\tau} \geq (t-1)C+\bar{\sigma}_{k,(t-1)\bar{S}+1}-\bar{\omega}_{kt} \qquad \begin{array}{l} k=1,\ldots,\bar{L} \\ t=1,\ldots,T \end{array} \tag{3.45}$$

O fim do tempo de troca de um lote deverá ocorrer em um único micro-período. Além disso, o xarope que é armazenado em um tanque está disponível para ser usado apenas quando o tempo de troca do tanque é completado:

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \bar{x}^E_{ks\tau}=1 \qquad \begin{array}{l} k=1,\ldots,\bar{L} \\ t=1,\ldots,T \\ s=(t-1)\bar{S}+1,\ldots,t\cdot\bar{S} \end{array} \tag{3.46}$$

$$\sum_{j=1}^{J}\bar{q}_{jks\tau} \leq \overline{Q}_k\bar{x}^E_{ks\tau} \qquad \begin{array}{l} k=1,\ldots,\bar{L} \\ t=1,\ldots,T \\ s=(t-1)\bar{S}+1,\ldots,t\cdot\bar{S} \\ \tau=(t-1)T^m+1,\ldots,t\cdot T^m \end{array} \tag{3.47}$$

O início do tempo de troca de um tanque também deverá ocorrer em um único micro-período:

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \bar{x}^B_{ks\tau}=1 \qquad \begin{array}{l} k=1,\ldots,\bar{L} \\ t=1,\ldots,T \\ s=(t-1)\bar{S}+2,\ldots,t\cdot\bar{S} \end{array} \tag{3.48}$$

$$\sum_{\tau=(t-2)T^m+1}^{t\cdot T^m} \bar{x}^B_{k,(t-1)\bar{S}+1,\tau}=1 \qquad \begin{array}{l} k=1,\ldots,\bar{L} \\ t=2,\ldots,T \end{array} \tag{3.49}$$

$$\sum_{\tau=1}^{T^m}\bar{x}^B_{k1\tau}=1 \qquad k=1,\ldots,\bar{L} \tag{3.50}$$

O intervalo de tempo no qual a troca de um lote é programada será positivo, se e somente se, o lote é efetivamente atribuído ao tanque:

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m}\tau\bar{x}^E_{ks\tau}-\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m}\tau\bar{x}^B_{ks\tau}$$

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau \bar{x}^E_{k,(t-1)\bar{S}+1,\tau} - \sum_{\tau=(t-2)T^m+1}^{t\cdot T^m} \tau \bar{x}^B_{k,(t-1)\bar{S}+1,\tau} \leq T^m \bar{u}_{k,(t-1)\bar{S}+1} \quad \begin{matrix} k=1,\ldots,\bar{L} \\ t=2,\ldots,T \end{matrix} \tag{3.52}$$

$$\sum_{\tau=1}^{T^m} \tau \bar{x}^E_{k1\tau} - \sum_{\tau=1}^{T^m} \tau \bar{x}^B_{k1\tau} \leq T^m \bar{u}_{k1} \quad k=1,\ldots,\bar{L} \tag{3.53}$$

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau \bar{x}^B_{ks\tau} - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau \bar{x}^E_{ks\tau} \leq \bar{u}_{ks} \quad \begin{matrix} k=1,\ldots,\bar{L} \\ t=1,\ldots,T \\ s=(t-1)\bar{S}+2,\ldots,t\cdot\bar{S} \end{matrix} \tag{3.54}$$

$$\sum_{\tau=(t-2)T^m+1}^{t\cdot T^m} \tau \bar{x}^B_{k,(t-1)\bar{S}+1,\tau} - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau \bar{x}^E_{k,(t-1)\bar{S}+1,\tau} \leq \bar{u}_{k,(t-1)\bar{S}+1} \quad \begin{matrix} k=1,\ldots,\bar{L} \\ t=2,\ldots,T \end{matrix} \tag{3.55}$$

$$\sum_{\tau=1}^{T^m} \tau \bar{x}^B_{k1\tau} - \sum_{\tau=1}^{T^m} \tau \bar{x}^E_{k1\tau} \leq \bar{u}_{k1} \quad k=1,\ldots,\bar{L} \tag{3.56}$$

Para evitar redundâncias, os lotes vazios nos tanques também são programados à direita do final dos lotes anteriores. Além disso, os lotes vazios serão os últimos lotes dentro do macro-período:

$$\sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau \bar{x}^B_{ks\tau} - \sum_{\tau=(t-1)T^m+1}^{t\cdot T^m} \tau \bar{x}^E_{k,s-1,\tau} \leq T^m \bar{u}_{ks} \quad \begin{matrix} k=1,\ldots,\bar{L} \\ t=1,\ldots,T \\ s=(t-1)S+2,\ldots,t\cdot S \end{matrix} \tag{3.57}$$

$$\sum_{\tau=(t-2)T^m+1}^{t\cdot T^m} \tau \bar{x}^B_{k,(t-1)S+1,\tau} - ((t-1)T^m+1) \leq T^m \bar{u}_{k,(t-1)S+1} \quad \begin{matrix} k=1,\ldots,\bar{L} \\ t=2,\ldots,T \end{matrix} \tag{3.58}$$

$$\sum_{\tau=1}^{T^m} \tau \bar{x}^B_{k1\tau} - 1 \leq T^m \bar{u}_{k1} \quad k=1,\ldots,\bar{L} \tag{3.59}$$

$$\bar{u}_{ks} \geq \bar{u}_{k,s+1} \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ t = 1, \ldots, T \\ s = (t-1)\bar{S} + 1, \ldots, t \cdot \bar{S} - 1 \end{array} \tag{3.60}$$

Sem perda de generalidade, consideramos que o tempo de troca entre xaropes é um inteiro múltiplo da capacidade do micro-período. Assim, o intervalo de tempo programado para início e fim do tempo de troca de um lote será exatamente igual ao tempo de troca envolvido:

$$\begin{array}{l} C^m \bar{u}_{ks} + \sum\limits_{\tau=(t-1)T^m+1}^{t \cdot T^m} C^m \tau \bar{x}^E_{ks\tau} \\ - \sum\limits_{\tau=(t-1)T^m+1}^{t \cdot T^m} C^m \tau \bar{x}^B_{ks\tau} = \bar{\sigma}_{ks} \end{array} \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ t = 1, \ldots, T \\ s = (t-1)\bar{S} + 2, \ldots, t \cdot \bar{S} \end{array} \tag{3.61}$$

$$\begin{array}{l} C^m \bar{u}_{k,(t-1)\bar{S}+1} + \sum\limits_{\tau=(t-1)T^m+1}^{t \cdot T^m} C^m \tau \bar{x}^E_{k,(t-1)\bar{S}+1,\tau} \\ - \sum\limits_{\tau=(t-2)T^m+1}^{t \cdot T^m} C^m \tau \bar{x}^B_{k,(t-1)\bar{S}+1,\tau} = \bar{\sigma}_{k,(t-1)\bar{S}+1} \end{array} \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ t = 2, \ldots, T \end{array} \tag{3.62}$$

$$\begin{array}{l} C^m \bar{u}_{k1} + \sum\limits_{\tau=1}^{T^m} C^m \tau \bar{x}^E_{k1\tau} \\ - \sum\limits_{\tau=1}^{T^m} C^m \tau \bar{x}^B_{k1\tau} = \bar{\sigma}_{k1} - \bar{\omega}_{k1} \end{array} \qquad k = 1, \ldots, \bar{L} \tag{3.63}$$

As restrições a seguir garantem que um tanque poderá ser reabastecido apenas quando estiver vazio:

$$\sum_{j=1}^{\bar{J}} \bar{I}_{jk,\tau-1} \leq \overline{Q}_k((1 - \bar{x}^B_{ks\tau}) + (1 - \bar{u}_{ks})) \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ t = 1, \ldots, T-1 \\ s = (t-1)\bar{S} + 1, \ldots, t \cdot \bar{S} + 1 \\ \tau = (t-1)T^m + 1, \ldots, t \cdot T^m \end{array} \tag{3.64}$$

$$\sum_{j=1}^{\bar{J}} \bar{I}_{jk,\tau-1} \leq \overline{Q}_k((1 - \bar{x}^B_{ks\tau}) + (1 - \bar{u}_{ks})) \qquad \begin{array}{l} k = 1, \ldots, \bar{L} \\ s = (T-1)\bar{S} + 1, \ldots, T \cdot \bar{S} \\ \tau = (T-1)T^m + 1, \ldots, T \cdot T^m \end{array} \tag{3.65}$$

3.4 Conclusão

O modelo matemático apresentado combinou aspectos do PGDPL e do PDLC para descrever as diversas restrições presentes no PCDLPP. Primeiro, os macro-períodos possuem um número máximo de lotes para serem ocupados pelos itens. Dessa forma, a atribuição de produtos e xaropes às linhas e tanques e a seqüência de ocupação das linhas e tanques foi descrita no modelo através de restrições com variáveis indexadas nos lotes $s = 1, ..., S$ para linhas e $s = 1, ..., \overline{S}$ para tanques. Segundo, a discretização do horizonte em micro-períodos de tamanho fixo levou à sincronia no tempo do que era decidido nas linhas com o que era decidido nos tanques. Isso foi descrito por restrições envolvendo variáveis de decisão indexadas nos micro-períodos $\tau = (t-1)T^m + 1, ..., tT^m$ com $t = 1, ..., T$. Os micro-períodos permitem estabelecer no tempo a ocupação das linhas e tanques. O modelo é bastante complexo dado o número elevado de parâmetros, variáveis e restrições. Até o momento, não encontramos abordagem similar na literatura. Modelos que descrevem problemas de dimensionamento e programação de lote acabam se enquadrando em um dos problemas descritos no capítulo 2. O modelo aqui proposto trata de forma interdependente dois problemas envolvendo dimensionamento e programação do lote. Devida à necessidade de uma sincronia no tempo entre os dois níveis envolvidos, acreditamos que o modelo proposto fornece um novo caminho em termos de modelagem para esse tipo de problema. Todavia, trata-se de um ponto de partida a partir do qual reformulações poderão levar a modelos capazes de representar melhor o problema em questão ou problemas similares.

Capítulo 4

Geração de instâncias

4.1 Introdução

Muitos autores utilizam instâncias existentes na literatura para avaliar seus modelos e métodos. Esse foi o caso de Meyr (2000) que adaptou instâncias definidas em Smith-Daniels and Smith-Daniels (1986) às restrições do PGDPL (Problema Geral de Dimensionamento e Programação de Lote). As mudanças realizadas pelo autor permitiram que:

1. o estado de preparo da máquina (*machine setup state*) fosse conservado após períodos ociosos de produção;

2. o tempo de preparo ocorresse apenas entre itens diferentes;

3. o custos de troca entre itens diferentes fosse duas vezes maior que entre mesmos itens.

Fleischmann and Meyr (1997) fez o mesmo ao solucionar o PDDPL (Problema Discreto de Dimensionamento e Programação de Lote) com custos de troca dependentes da seqüência. O autor adaptou instâncias do PDLC (Problema de Dimensionamento de Lote Capacitado) apresentadas em Thizy and van Wassenhove (1985), dividindo seus períodos de tempo em micro-períodos de mesma capacidade. O autor também utilizou a matriz de custos de preparo independentes da seqüência apresentada no PDLC, mas gerou outras 5 matrizes com custos de troca dependentes da seqüência.

Não encontramos na literatura conjuntos de instâncias que reunissem todas as características presentes no PCDLPP proposto nesta tese. Esse é o principal motivo pelo qual não utilizamos instâncias existentes na literatura. As mudanças necessárias para adequação de tais instâncias ao PCDLPP não seriam tão diretas como as realizadas nos trabalhos mencionados.

Os números de variáveis e restrições presentes no PCDLPP crescem rapidamente em função de alguns parâmetros como número de linhas (L), tanques ($\bar{L}$), produtos (J), xaropes ($\bar{J}$), macro-períodos

(T), micro-períodos (T_m), lotes nas linhas (S) e lotes nos tanques ($\bar{S}$). A adaptação de instâncias de outros problemas poderia levar a modelos com um número elevado de variáveis binárias e, conseqüêntemente, de difícil resolução dentro de um tempo computacional aceitável.

Por isso, optou-se pelo desenvolvimento de um método para geração de instâncias do PCDLPP que nos permitesse controlar a dimensão dos parâmetros envolvidos. Dividimos as instâncias geradas pelo método em dois grupos: instâncias de pequena dimensão e instâncias de elevada dimensão. O número de variáveis binárias presentes no modelo matemático correspondente à cada instância foi o critério utilizado para separá-las nesses dois grupos.

O presente capítulo descreve o método gerador de instâncias desenvolvido para o PCDLPP. Também serão apresentados resultados computacionais que nos ajudaram a determinar os valores de alguns parâmetros utilizados na geração das instâncias.

4.2 Método gerador de instâncias

O método gerador de instâncias basicamente obedece ao seguinte algoritmo:

Passo1: Recebe parâmetros:

$$\{J, \bar{J}, L, \bar{L}, T, T^m, C^m\}$$
$$\{h_j, \overline{h}_j, v_{jl}, \overline{v}_{jk}, \overline{Q}_k, \underline{Q}_k\}$$
$$\{minDem, maxDem, minp_{jl}, maxp_{jl}, minr_{ji}, maxr_{ji}\}$$
$$\{min\sigma'_{ijl}, max\sigma'_{ijl}, min\bar{\sigma'}_{ijk}, max\bar{\sigma'}_{ijk}\}$$

Passo2: Gera tempos de troca nas linhas (σ'_{ijl}) e tanques ($\bar{\sigma'}_{ijk}$), tempo de processamento (p_{jl}) e fator de conversão (r_{ji}):

$$\sigma'_{ijl} \in [min\sigma'_{ijl}, max\sigma'_{ijl}],\ \bar{\sigma'}_{ijk} \in [min\bar{\sigma'}_{ijk}, max\bar{\sigma'}_{ijk}]$$
$$p_{jl} \in [minp_{jl}, maxp_{jl}],\ r_{ji} \in [minr_{ji}, maxr_{ji}]$$

Passo3: Calcula os custos de troca:

$$s_{ij} = f\sigma'_{ijl}$$
$$\overline{s}_{ij} = f\overline{\sigma'}_{ijk}$$

Passo4: Gera demandas para os produtos:

$$d_{jt} \in [minDem, maxDem]$$

Passo5: Calcula o tempo de processamento médio das demandas ($\overline{TP}$).

Passo6: Seja C a capacidade disponível no macro período.

Passo6.1: Se $0,8C <= \overline{TP} <= 1,2C$, vá para o Passo7

Passo6.2: Caso contrário, retorne ao Passo4

Passo7: Cria instância com os parâmetros gerados.

O método recebe vários parâmetros no Passo1. O primeiro conjunto de parâmetros contém o número de produtos (J) e xaropes ($\overline{J}$), o número de linhas (L) e tanques ($\overline{L}$), o número de macro (T) e micro-períodos (T^m) e a capacidade disponível em cada micro-período (C^m). O próximo conjunto de parâmetro apresenta os custos de estoque dos produtos (h_j) e xaropes ($\overline{h}_j$), os custos de produção dos produtos (v_{jl}) e xaropes ($\overline{v}_{jk}$), respectivamente, nas linhas e tanques. Também são recebidos os valores das capacidades máxima e mínima dos tanques ($\overline{Q}_k$ e $\underline{Q}_k$). Os próximos parâmetros obtidos se referem aos valores máximos e mínimos que estabelecem os intervalos onde as demandas, os tempos de processamento, os fatores de conversão e os tempos de troca são gerados.

No Passo2, os tempos de troca nas linhas (σ'_{ijl}) e tanques ($\bar{\sigma'}_{ijk}$), os tempos de processamento (p_{jl}) e os fatores de conversão (r_{ji}) são gerados de modo uniforme dentro dos intervalos estabelecidos. Uma vez determinado os tempos de troca para linhas e tanques, os custos de troca são calculados no Passo3. Esses valores são obtidos a partir dos tempos de troca através do fator de conversão f. As razões para se obter os custos de troca dessa maneira e os valores adotados para f são apresentados na seção 4.4.

As demandas em cada período T são geradas no Passo4. Procuramos criar instâncias onde as demandas possam ser atendidas dentro da capacidade disponível. Os valores já definidos para T e T^m limitam a capacidade em $C = T^m \text{x} C^m$ por macro-período t. Por isso, um valor para o tempo de processamento médio ($\overline{TP}$) das demandas criadas é calculado no Passo5. As fórmulas que determinam $\overline{TP}$ são estabelecidas na próxima seção. Uma vez determinado $\overline{TP}$, as demandas são incorporadas à instância apenas se a condição do Passo6 for atendida:

$$0,8C <= \overline{TP} <= 1,2C$$

Considerou-se uma tolerância dentro da qual $\overline{TP}$ deve estar para se garantir um certo nível de utilização da capacidade disponível. Caso essa condição seja atendida, a instância é gerada (Passo7). Caso contrário, retorna-se ao Passo4 e um novo conjunto de demandas é criado.

Não há garantias de que uma instância factível será gerada. Os passos do algoritmo proposto apenas permitem a geração de instância seguindo alguns critérios. Na prática, a adoção desses critérios permitiu a obtenção de um número maior de instâncias factíveis quando comparado a um procedimento de simples geração aleatória dos parâmetros. Consideramos instâncias factíveis nesse contexto as soluções que atendam todas as restrições do modelo matemático e satisfaçam as demandas envolvidas ($q_j^0 = 0, \forall j \in J$).

4.2.1 Instâncias de pequena dimensão

Os valores usados nos parâmetros do Passo1 para criar instâncias consideradas de pequena dimensão foram:

- $L = \{2, 3, 4\}$ e $\bar{L} = \{2, 3\}$.
- $J = \{2, 3, 4\}$ e $\bar{J} = \{1, 2\}$.
- $S = \{2, 3\}$ e $\bar{S} = \{2\}$.
- $T = \{1, 2, 3, 4\}$, $T^m = 5$ e $C^m = 1h$.
- $minDem = 500u$ e $maxDem = 10000u$, u: unidade de produto.
- $h_j = \overline{h}_j = 1$ (\$/u).
- $v_{jl} = \bar{v}_{jk} = 1$ (\$/u).
- $\overline{Q}_k = 5000l$ e $\underline{Q}_k = 1000l$, l: litros de xarope.

Parâmetros como T^m, C^m, h_j, $\overline{h}_j$, $\overline{Q}_k$ e $\underline{Q}_k$ receberam valores que não são alterados nos diferentes conjutos de instâncias criados. Diversas variáveis e restrições no modelo matemático são indexadas nos valores do parâmetro T^m, por isso, a escolha de $T^m = 5$ apresentou razoável impacto sobre o número de variáveis (binárias ou não) e de restrições do modelo matemático correspondente.

O tempo de processamento combinado com o tempo de preparo de um produto em geral ocupam mais de 1 unidade da capacidade disponível, pelos valores aqui utilizados. Ao arbitrarmos $C^m = 1$, procuramos evitar que períodos ociosos ocorressem, já que dois produtos não podem ser iniciados em um mesmo micro-período (hipótese do modelo matemático).

A figura 4.1 exemplifica essa idéia. A situação (i) na figura aproveita melhor a capacidade disponível no micro-período para produção de A, B e C. Por outro lado, o produto C já não consegue ser programado dentro de um horizonte de 6 unidades de tempo quando $C^m = 2$.

Os custos de estoque nas linhas (h_j) e tanques ($\overline{h}_j$) foram fixados em 1 unidade financeira por unidade de produto , ou seja, são proporcionais à quantidade de produtos armazenada nos períodos onde eles incidem. Os custos dos produtos(v_{jl}) e xaropes ($\overline{v}_{jk}$) também são de 1 unidade financeira por unidade de produto.

$\overline{Q}_k = 5000l$ foi adotado após testes usando a fórmula para o cálculo do tempo de processamento médio $\overline{TP}$ apresentada na próxima seção. O valor escolhido para $\overline{Q}_k$ permite que no cálculo de $\overline{TP}$ as demandas geradas atendam um certo nível de utilização da capacidade disponível. O mesmo critério foi utilizado para determinar os valores de $minDem$ e $maxDem$. $\overline{Q}_k = 5000l$ não confere ao tanque

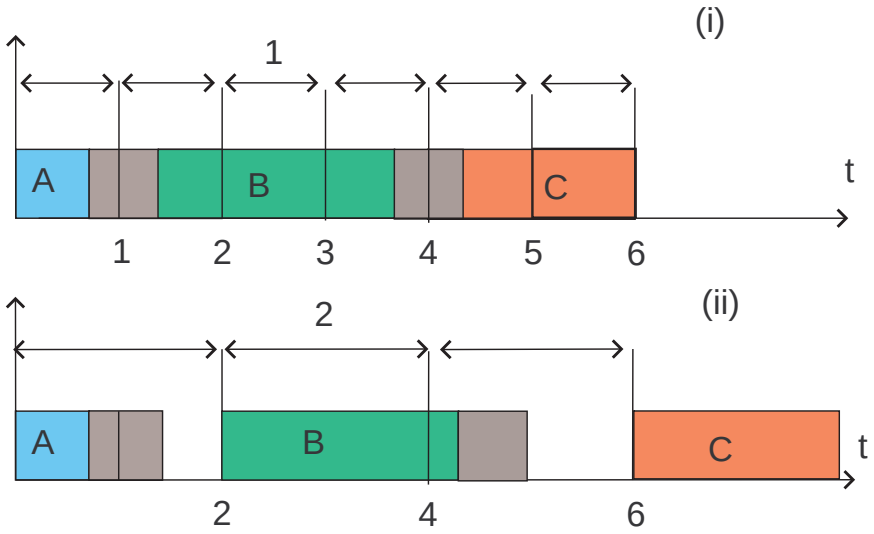

Fig. 4.1: Tamanho dos Micro-períodos: (i).$C^m = 1$; (ii).$C^m = 2$.

capacidade infinita, permitindo a ocorrência de tempos de troca pelo reabastecimento do tanque. Para a capacidade mínima foi adotado o valor de um quinto da capacidade máxima, ou seja, $\underline{Q}_k = 1000l$. Assim, o valor mínimo não é irrisório tornando efetiva a restrição de ocupação mínima de um tanque.

Um conjunto de 10 instâncias foi criado para cada combinação dos parâmetros LxJx$T = 3$x3x$4 = 36$, ou seja, 360 instâncias no total. Alguns parâmetros foram fixados ao invés de combinados. A tabela 4.1 apresenta os diferentes conjuntos de instância de pequena dimensão, considerando-se apenas as Lx$J = 3$x$3 = 9$ combinações.

L	$\overline{L}$	J	$\overline{J}$	S	$\overline{S}$
2	2	2	1	2	2
2	2	3	2	3	2
2	2	4	2	3	2
3	3	2	1	2	2
3	3	3	2	3	2
3	3	4	2	3	2
4	4	2	1	2	2
4	4	3	2	3	2
4	4	4	2	3	2

Tab. 4.1: Combinação de parâmetros em cada conjunto de instâncias

Observe que o número de xaropes ($\bar{J}$), o número de tanques ($\bar{L}$), o número de lotes nos tanques ($\bar{S}$) e lotes nas linhas (S) são fixados e não combinados com L e J. Por exemplo, fixou-se $\bar{L} = 2$ e $S = \bar{S} = 2$ quando $L = 2$ e $\bar{L} = 3$, $S = 3$ e $\bar{S} = 2$ quando $L = 3$ ou 4.

Adotou-se tal procedimento por dois motivos. Primero, uma combinação muito ampla de parâmetros levaria a um número excessivo de possíveis instâncias a serem geradas. Segundo, parâmetros como S e $\bar{S}$ repercutem diretamente no número de variáveis e restrições do modelo. Ao fixar tais valores, evitamos que as instâncias se tornem de elevada dimensão de acordo com os critérios adotados nesse trabalho.

A tabela 4.2 apresenta os 36 conjuntos de instâncias obtidos com as diferentes combinações de parâmetros e informações sobre o número de variáveis e restrições presentes no respectivo modelo matemático.

4.2.2 Instâncias de elevada dimensão

Alguns valores dos parâmetros foram alterados para se gerar as instâncias de elevada dimensão:

- $L = \{5, 8\}$ e $\bar{L} = \{5, 6\}$.
- $J = \{10, 15\}$ e $\bar{J} = \{5, 8\}$.
- $S = \{5, 8\}$ e $\bar{S} = \{3, 4\}$.
- $T = \{4, 8, 12\}$ e $T^m = \{10\}$.

Cresce o número de produtos, linhas e macro-períodos nas instâncias de elevada dimensão. Logo, outros parâmetros como número de lotes por período (S e $\bar{S}$) e número de micro-períodos (T^m) precisaram ser ajustados. O critério adotado para escolha desses valores foi o mesmo utilizado nas instâncias de pequena dimensão.

A tabela 4.3 apresenta um total de 12 conjuntos de instâncias de elevada dimensão com os parâmetros iniciais e o respectivo número de variáveis binárias nos modelos correspondentes. Para cada um desses conjuntos, 10 instâncias foram geradas.

O método de geração de instâncias no Passo2 determina aleatoriamente valores para o tempo de preparo seqüência-dependente nas linhas (σ'_{ijl}) e tanques ($\sigma^{\bar{l}}_{ijk}$), o tempo de processamento dos produtos em cada linha (p_{jl}) e o fator de conversão dos xaropes em bebidas (r_{ji}). A tabela 4.4 lista os intervalos de valores, onde U[a,b] indica que foram gerados de forma uniforme no intervalo [a,b].

Os valores para o tempo de preparo nas linhas oscilam entre meia e uma hora. Os tempos de preparo nos tanques variam entre uma e duas horas. Os tempos de processamento variam entre 1000 e 2000 produtos por hora. Os fatores de conversão dos xaropes em produtos estão entre 0,3 e 3 litros de xarope por unidade do produto (bebida). Todos esses limitantes foram baseados em alguns dados fornecidos por uma indústria de bebidas.

	Parâmetros							Modelo		
L	J	T	$\overline{L}$	$\overline{J}$	S	$\overline{S}$	T^m	Var. Binárias	Var. Contínuas	Restrições
2	2	1	2	1	2	2	5	100	263	281
2	3	1	2	2	3	2	5	136	499	454
2	4	1	2	2	3	2	5	142	615	512
3	2	1	2	1	2	2	5	150	452	419
3	3	1	2	2	3	2	5	204	880	673
3	4	1	2	2	3	2	5	213	1098	764
4	2	1	2	1	2	2	5	176	555	498
4	3	1	2	2	3	2	5	246	1100	821
4	4	1	2	2	3	2	5	258	1390	924
2	2	2	2	1	2	2	5	210	533	575
2	3	2	2	2	3	2	5	282	1004	919
2	4	2	2	2	3	2	5	294	1235	1039
3	2	2	2	1	2	2	5	315	916	862
3	3	2	2	2	3	2	5	423	1771	1368
3	4	2	2	2	3	2	5	441	2206	1552
4	2	2	2	1	2	2	5	367	1122	1013
4	3	2	2	2	3	2	5	507	2211	1641
4	4	2	2	2	3	2	5	531	2790	1865
2	2	3	2	1	2	2	5	320	803	863
2	3	3	2	2	3	2	5	574	2014	1835
2	4	3	2	2	3	2	5	598	2475	2075
3	2	3	2	1	2	2	5	480	1380	1303
3	3	3	2	2	3	2	5	642	2662	2071
3	4	3	2	2	3	2	5	669	3314	2350
4	2	3	2	1	2	2	5	558	1689	1536
4	3	3	2	2	3	2	5	768	3322	2466
4	4	3	2	2	3	2	5	804	4190	2799
2	2	4	2	1	2	2	5	430	1073	1163
2	3	4	2	2	3	2	5	574	2014	1827
2	4	4	2	2	3	2	5	598	2475	2075
3	2	4	2	1	2	2	5	645	1844	1752
3	3	4	2	2	3	2	5	861	3553	2764
3	4	4	2	2	3	2	5	897	4422	3140
4	2	4	2	1	2	2	5	749	2256	2039
4	3	4	2	2	3	2	5	1029	4433	3335
4	4	4	2	2	3	2	5	1077	5590	3735

Tab. 4.2: Instâncias de pequena dimensão

4.3 Cálculo do Tempo de Processamento Médio

A abordagem adotada para o cálculo de $\overline{TP}$ é inspirada nos trabalhos de Berreta (1997) e Toledo (1998) que também tratam do dimensionamento de lotes em processadores dispostos em paralelo. As

	Parâmetros							Modelo		
L	J	T	$\overline{L}$	$\overline{J}$	S	$\overline{S}$	T^m	Var. Binárias	Var. Contínuas	Restrições
5	10	4	5	5	5	3	10	4810	71961	23857
5	15	4	6	8	8	4	10	8724	208172	65166
8	10	4	5	5	5	3	10	6670	110553	33778
8	15	4	6	8	8	4	10	12180	321272	94963
5	10	8	5	5	5	3	10	9670	143961	47549
5	15	8	6	8	8	4	10	17508	416388	132258
8	10	8	5	5	5	3	10	13390	221145	68114
8	15	8	6	8	8	4	10	24420	642588	188687
5	10	12	5	5	5	3	10	14530	215961	70657
5	15	12	6	8	8	4	10	26292	624604	197318
8	10	12	5	5	5	3	10	20110	331737	101914
8	15	12	6	8	8	4	10	36660	963904	286667

Tab. 4.3: Instâncias de elevada dimensão

Parâmetros	valores	Observações
σ'_{ijl}	$U[0.5, 1]$	tempo de preparo nas linhas
$\overline{\sigma'}_{ijk}$	$U[1, 2]$	tempo de preparo nos tanques
p_{jl}	$U[1000, 2000]$	tempo de processamento
r_{ji}	$U[0.3, 3]$	fator de conversão

Tab. 4.4: Parâmetros adotados

autoras utilizam uma política lote-por-lote para determinar a capacidade de cada processador dentro de cada período. Nessa política, a capacidade necessária é calculada considerando que a produção em um período deverá atender toda a demanda correspondente. As demandas são divididas entre os processadores e determina-se a capacidade média sobre o número de períodos e máquinas. As subseções a seguir apresentam passo a passo o cálculo de $\overline{TP}$.

4.3.1 Cálculo do tempo de processamento médio entre os macro-períodos

A fórmula abaixo calcula o tempo de processamento total em uma linha l dentro de um macro-período t (TP_{lt}). A demanda de um produto j em t é dividida entre as linhas capazes de produzi-la ($\frac{d_{jt}}{|L_j|}$) e temos:

$$TP_{lt} = \sum_{j=1}^{J}(p_{jl}\frac{d_{jt}}{|L_j|} + s_{jl}) \tag{4.1}$$

onde,

TP_{lt} : Tempo de processamento total consumido pela linha l na produção de $\frac{d_{jt}}{|L_j|}$ quando $l \in L_j$ no macro-período t, $\forall j \in J$.

d_{jt} : Demanda do produto $j \in J_l$ no macro-período t.

p_{jl} : tempo de processamento de uma unidade do produto j na linha l.

L_j : Conjunto de linhas nas quais o produto j pode ser produzido.

J_l : Conjunto de produtos que podem ser produzidos pela linha l.

s_{jl} : tempo de troca médio entre os demais produtos $i \in J_l$ e o produto j na linha l.

$$s_{jl} = \frac{\sum_{i=1}^{J} s_{ijl}}{|J_l|} \tag{4.2}$$

A partir de TP_{lt}, o tempo de processamento médio entre as linhas em cada macro-período é calculado. O número de produtos $|J_l|$ em cada linha atua como peso no cálculo desta média ponderada.

$$TP_t = \frac{\sum_{l=1}^{L} |J_l| TP_{lt}}{\sum_{l=1}^{L} |J_l|} \tag{4.3}$$

onde,

TP_t : Média ponderada do tempo de processamento das linhas em t.

A partir de TP_t, o tempo de processamento médio entre os macro-períodos é calculado. O número de produtos com demanda positiva em t ($|J_t|$) atua como peso.

$$TP = \frac{\sum_{t=1}^{T} |J_t| TP_t}{\sum_{t=1}^{T} |J_t|} \tag{4.4}$$

onde,

TP : Média ponderada do tempo de processamento nos macro-períodos.

J_t : Conjunto de produtos com demanda positiva no macro-período t.

4.3.2 Cálculo do tempo de preparo médio dos xaropes

O valor obtido para TP considera o tempo de preparo entre produtos nas linhas de produção, além do tempo de processamento. Todavia, o tempo de preparo entre xaropes nos tanques poderá repercutir nas linhas de produção, já que sem xarope no tanque a linha não consegue produzir. Dependendo do tipo de programação para linhas e tanques, essa situação poderia ser evitada.

Por exemplo, considere a programação para linhas e tanques apresentada na figura 4.2. Três pro-

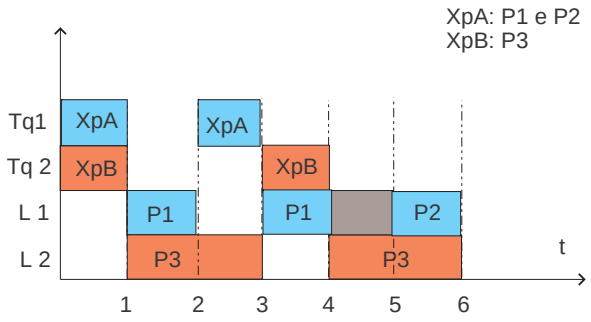

Fig. 4.2: Caso1: tempo de preparo dos xaropes repercutindo nas linhas

dutos e dois xaropes estão programandos, onde o xarope XpA atende aos produtos P1 e P2 e o xarope XpB atende P3. Nesse caso, a ocupação do tanque Tq2 por XpB faz com que o reabastecimento do tanque Tq1 com XpA provoque a interrupção da produção de P1 na linha L1. Observe que a troca de produtos em L1 ocorre apenas mais tarde. O cenário seria diferente caso a produção de P3 não ocoresse como pode ser visto na figura 4.3. A não ocorrência de XpB em Tq2, permite a ocupação

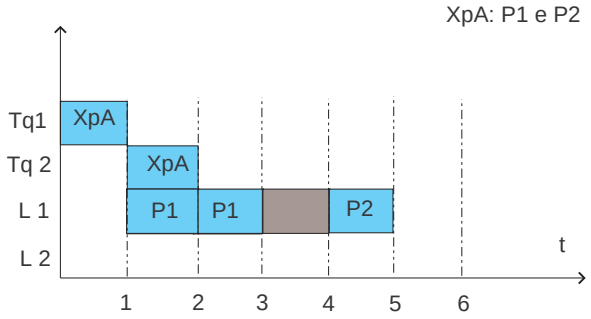

Fig. 4.3: Caso2: tempo de preparo dos xaropes repercutindo nas linhas

do tanque por XpA (supondo que XpA possa ser armazenado em Tq1 e Tq2). Isso faz com que o tempo de troca entre xaropes deixe de repercutir na linha L1, ocorrendo apenas o tempo de troca entre produtos (supondo que P1 e P2 sejam produzidos apenas por L1).

Se não considerarmos o tempo de preparo dos xaropes no valor do tempo de processamento médio $\overline{TP}$, as demandas geradas podem deixar pouca capacidade disponível para o tempo de preparo dos xaropes e a instância criada seria infactível. Testes realizados demonstraram que sem a incorporação do tempo de preparo médio dos xaropes TPX, crescia o número de instâncias infactíveis.

Por isso, o valor de TPX será considerado. A quantidade de xarope que é demandada em cada macro-período será dividida entre os tanques que possam produzi-la. Dado um xarope, o total armazenado em cada tanque é dividido pela sua capacidade máxima. O maior inteiro menor que o valor da divisão anterior determina o número de trocas envolvendo um mesmo xarope. O tempo de troca

médio entre outros xaropes e o xarope em questão também é considerado.

$$nS_j = \frac{D_{jt}}{|\bar{L}_j|\bar{Q}_k} \tag{4.5}$$

$$TPX_{kt} = \sum_{j=1}^{\bar{J}} (\lfloor nS \rfloor \bar{s}_{jjk} + \bar{s_{jk}}) \tag{4.6}$$

onde,

nS_j : Número de trocas envolvendo o mesmo xarope j no tanque k.

TPX_{kt} : Tempo total de preparo gasto no tanque k quando $k \in \bar{L}_j$.

D_{jt} : demanda por xarope no macro-período t, calculada segundo a fórmula abaixo, onde r_{ji} é a quantidade de xarope j necessária para produzir uma unidade do produto i.

$$D_{jt} = \sum_{i=1}^{J} r_{ji} d_{it} \tag{4.7}$$

$\bar{L}_j$: conjunto de tanques nos quais o xarope j pode ser armazenado.

$\bar{Q}_k$: capacidade máxima do tanque k.

$\bar{s}_{jjk}$: tempo de preparo gasto na troca de um mesmo xarope no tanque k.

$\bar{s}_{jk}$: tempo de preparo médio entre outros xaropes e o xarope j no tanque k.

$$\mu_{jk} = \frac{\sum_{i=1}^{\bar{J}} \bar{s}_{ijl}}{|\bar{J}_k|} \tag{4.8}$$

$$\bar{s}_{jk} = \lceil \mu_{jk} \rceil \tag{4.9}$$

$\bar{J}_k$: Conjunto de xaropes capaz de ser armazenado no tanque k.

$\lfloor x \rfloor$: Função piso. Maior inteiro menor ou igual a x.

$\lceil x \rceil$: Função teto. Menor inteiro maior ou igual a x.

Assim, o tempo médio de troca entre xaropes em um macro-período t será:

$$TPX_t = \frac{\sum_{k=1}^{\bar{S}} |\bar{J}_k| TPX_{kt}}{\sum_{k=1}^{\bar{S}} |\bar{J}_k|} \tag{4.10}$$

onde,

TPX_t : Média ponderada do tempo de troca em um macro-período t.

$\bar{J}_k$: Conjunto de xaropes capaz de ser armazenado no tanque k.

O tempo médio de troca entre xaropes, considerando os macro-períodos será:

$$T\bar{P}X = \frac{\sum_{t=1}^{T} |\bar{D}_t| TPX_t}{\sum_{t=1}^{T} |\bar{D}_t|} \tag{4.11}$$

onde,

TP : Média ponderada do tempo de troca entre xaropes.

$\bar{D}_t$: Conjunto de xaropes cuja demanda é positiva em t ($D_{jt} > 0$).

4.3.3 Cálculo do tempo de preparo médio

Por último, chegamos à fórmula para determinação do tempo de processamento médio $\overline{TP}$.

$$\overline{TP} = TP + TPX \tag{4.12}$$

com TP e TPX obtidos de acordo com o exposto nas seções anteriores.

4.4 Determinação dos custos de preparo para linhas e tanques

A determinação destes custos foi baseada em uma abordagem apresentada por Haase and Kimms (2000). Os autores solucionam um PDLC com custos e tempos de troca dependentes da seqüência. Os tempos de troca st_{ij} com $i \neq j$ são aleatoriamente gerados no intervalo [2,10] e $st_{jj} = 0$. Os custos de troca são proporcionais ao tempo de troca de acordo com a seguinte expressão $sc = f_{sc} st_{ij}$, onde o parâmetro f_{sc} assume os valores $f_{sc} = 50$ e 500. Os autores avaliam o impacto resultante do uso desses diferentes valores de f_{sc} na determinação dos custos de troca a partir do tempo de troca.

Os custos de troca para linhas e tanques foram aqui determinados de forma semelhante :

$$s_{ijl} = f\sigma'_{ijl}$$
$$\overline{s}_{ijk} = f\overline{\sigma}'_{ijk}.$$

f é o parâmetro que estabelece a proporcionalidade entre custos e tempos de troca. Conforme já foi mencionado, $\sigma'_{ijl} \in U[0.5, 1]$ e $\bar{\sigma}'_{ijk} \in U[1, 2]$. O parâmetro f recebeu inicialmente dois possíveis valores: 1000 e 10000. Dessa forma, foram avaliados custos de troca que variam entre 500 e 2000 quando $f = 1000$ e entre 5000 e 20000 quando $f = 10000$.

O critério adotado para determinar qual o melhor valor para f se baseou na porcentagem que os custos de troca ocupam dentro do valor da função objetivo do modelo matemático:

$$\sum_{i=1}^{J}\sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{s=1}^{T\cdot S} s_{ijl}z_{ijls} + \sum_{j=1}^{J}\sum_{t=1}^{T} h_j I_{jt} + \sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{s=1}^{T\cdot S} v_{jl}q_{jls}$$
$$+\sum_{i=1}^{J}\sum_{j=1}^{J}\sum_{k=1}^{\bar{L}}\sum_{s=1}^{T\cdot \bar{S}} \bar{s}_{ijk}\bar{z}_{ijks} + \sum_{j=1}^{\bar{J}}\sum_{k=1}^{\bar{L}}\sum_{t=1}^{T} \bar{h}_j \bar{I}_{jk,t\cdot T^m} + \sum_{j=1}^{\bar{J}}\sum_{k=1}^{\bar{L}}\sum_{s=1}^{T\cdot \bar{S}} \bar{v}_{jk}\bar{q}_{jks}$$
$$+M\sum_{j=1}^{J} q_j^0 \tag{4.13}$$

Fleischmann (1990) utilizou critério semelhante para avaliar o impacto na função objetivo do uso de diferentes matrizes de custo de troca dependentes da seqüência.

A ferramenta computacional GAMS utilizando como solver o pacote Cplex versão 7.0 foi utilizada para avaliar os conjuntos de instâncias considerando $f = 1000$ e $f = 10000$. Os testes foram realizados em um Pentium IV com 2.8 GHz. Inicialmente, solucionamos conjuntos de instâncias com $T = 1$, $L = 2, 3, 4$ e $J = 2, 3$ e 4 e os resultados obtidos estão nas tabelas 4.5 e 4.6, onde

- J é o número de produtos e CPU é o tempo computacional gasto em segundos;
- $\%s$ e $\%\overline{s}$ indicam a porcentagem que o custo de troca das linhas e tanques ocupam na função objetivo;
- $\%s\overline{s}$ é a soma de $\%s$ e $\%\overline{s}$, ou seja, indica o percentual total de custos de troca na função objetivo;
- Os valores de CPU, $\%s$, $\%\overline{s}$ e $\%s\overline{s}$ correspondem à média obtida para as 10 instâncias de cada conjunto.

.

Observamos que o percentual dos custos de troca cresce quando o número de produtos aumenta em cada linha. Esse comportamento era esperado já que um maior número de produtos leva a um aumento no número de trocas nas linhas. Da mesma forma, o percentual diminui quando aumenta o número de linhas. Por exemplo, $\%s$ passa de 3,07% em $L = 2$ e $J = 2$ para 1,73% em $L = 4$ e $J = 2$. Trata-se de outro comportamento esperado já que um maior número de linhas permite melhor distribuição dos produtos (xaropes), diminuindo as trocas em uma mesma linha (tanque).

A porcentagem dos custos cresce quando $f = 10000$, devido ao maior valor de f, acompanhado por um aumento no tempo computacional. Isso pode ser melhor verificado nas tabelas 4.7 e 4.8 que apresentam os valores médios considerando todos os produtos em cada conjunto com $L = 2, 3$ e 4.

		L = 2		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	2,08	3,07	11,87	14,95
3	0,96	8,44	13,81	22,25
4	0,77	10,80	13,16	23,96
		L = 3		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	573,32	1,70	10,55	12,25
3	151,79	6,21	11,76	17,97
4	46,26	7,75	12,02	19,77
		L = 4		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	3068,64	1,73	10,63	12,36
3	989,39	3,31	12,86	16,17
4	1356,72	5,65	12,78	18,43

Tab. 4.5: $T = 1$, $f = 1000$

		L = 2		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	2,13	11,98	50,90	62,88
3	1,00	27,74	46,10	73,84
4	0,86	33,94	41,78	75,72
		L = 3		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	1100,69	7,48	50,14	57,62
3	348,33	23,69	44,80	68,49
4	69,58	27,37	43,51	70,88
		L = 4		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	3223,56	7,65	50,46	58,12
3	1709,09	13,06	52,51	65,57
4	1859,98	21,10	48,14	69,24

Tab. 4.6: $T = 1$, $f = 10000$

L	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	1,27	7,44	12,95	20,39
3	257,12	5,22	11,45	16,67
4	1804,91	3,56	12,09	15,65

Tab. 4.7: $T = 1$, $f = 1000$

L	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	1,33	24,55	46,26	70,81
3	506,20	19,51	46,15	65,66
4	2264,21	13,94	50,37	64,31

Tab. 4.8: $T = 1$, $f = 10000$

Observe que para $T = 1$, $\%s\overline{s}$ oscilou entre 20,39 na tabela 4.7 e 70,81 na tabela 4.8. O respectivo tempo computacional variou de $257,12$ para $506,20$. O mesmo comportamento ocorre quando $T = 2$ como pode ser observado nas tabelas 4.9 e 4.10. As porcentagens diminuem tanto quando $f = 1000$ e 10000 com $T = 2$ em relação aos conjuntos de instâncias com $T = 1$. Isso também ocorre com os valores médios apresentados nas tabelas 4.11 e 4.12.

		L=2		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	28,27	2,76	5,58	8,34
3	1703,08	5,69	8,38	14,08
4	1316,72	7,89	7,27	15,16
		L=3		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	3365,86	1,36	5,25	6,61
3	3600	2,52	6,67	9,19
4	3600	4,64	5,26	9,90
		L=4		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	3600	1,82	6,50	8,32
3	3600	4,38	4,82	9,20
4	3600	3,14	5,78	8,92

Tab. 4.9: $T = 2$, $f = 1000$

Podemos avaliar a mudança de patamar no valor da porcetagem total de custo de troca $\%s\overline{s}$ quando $T = 1, 2$ e $f = 1000, 10000$ na figura 4.4.

Observe que para $T = 1$ as porcetagens que estavam entre 15% e 21% quando $f = 1000$ ficam entre 64% e 71% quando $f = 10000$. Os valores diminuem quando $T = 2$. A porcentagem passam a ficar entre 8% e 13% quando $f = 1000$ e entre 44% e 53% quando f=10000.

Os valores fornecidos por $f = 1000$ nos parecem mais adequados para estabelecer uma proporcionalidade entre custos e tempos de troca. Custos de troca total ($\%s\overline{s}$) acima de 50% contém um certo exagero já que a função objetivo do problema não se preocupa em minimizar apenas esse tipo de custo. Da mesma forma, valores abaixo de 10% estariam de certa forma subestimando o peso do custo de troca no problema.

		L=2		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	178,26	10,22	30,26	40,48
3	1607,33	21,09	36,66	57,74
4	1548,09	30,71	28,48	59,19
		L=3		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	3600	2,01	33,52	35,53
3	3600	11,27	33,71	44,98
4	3600	11,45	30,12	41,57
		L=4		
J	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	3600	5,12	35,17	40,29
3	3600	19,05	23,91	42,96
4	3600	17,39	32,05	49,45

Tab. 4.10: $T = 2$, $f = 10000$

L	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
2	1016,02	5,45	7,08	12,53
3	3522,00	2,84	5,73	8,57
4	3600	3,11	5,70	8,81

Tab. 4.11: $T = 2$, $f = 1000$

L	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
L=2	1111,23	20,67	31,80	52,47
L=3	3600	8,24	32,45	40,69
L=4	3600	13,86	30,38	44,23

Tab. 4.12: $T = 2$, $f = 10000$

O próximo passo foi avaliar os custos em instâncias com elevada dimensão. Conforme já foi visto na tabela 4.3, o número de variáveis e restrições alcançam um patamar bem mais elevado quando comparado aos valores da tabela 4.2. Assim, a solução das instâncias de elevada dimensão usando o pacote computacional GAMS/Cplex fica inviável dentro de um tempo computacional razoável. Todos os testes realizados nessa etapa consideraram sempre 1 hora como limite máximo de tempo computacional. O GAMS/Cplex dificilmente consegue obter uma solução factível dentro desse tempo para instâncias de elevada dimensão.

Assim, para analisar o impacto dos custos de troca nessas instâncias, foi utilizado como método de resolução um Algoritmo Genético (AG). O AG será detalhado no capítulo 6 e aqui serviu apenas para avaliar o impacto dos parâmetros de custo de troca nas instâncias de elevada dimensão. Para

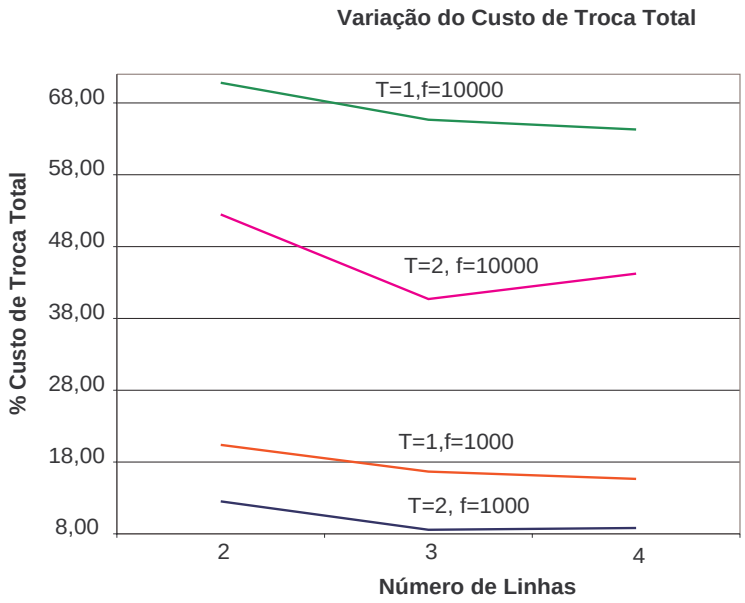

Fig. 4.4: $\%s\overline{s}$ quando $T = 1, 2$ e $L = 2, 3, 4$

			L=5		
J	T	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
10	4	745,74	5,51	11,59	17,10
15	4	1003,49	6,76	11,86	18,63
10	8	1132,24	6,32	14,56	20,88
10	12	1194,34	6,93	14,05	20,98

Tab. 4.13: $L = 5$, $f = 1000$

cada instância fornecida ao AG, o método foi executado três vezes durante $1200s.$ em cada execução. Os resultados obtidos usando AG para $f = 1000$ e $f = 10000$ estão nas tabelas 4.13 e 4.14.

Os resultados nos mostram que os valores para custo de troca total quando $f = 1000$ oscilam em torno de 20%. Os mesmos valores ficam entre 60% e 70% quando $f = 10000$. O crescimento no valor dos custos não levou a um aumento considerável no tempo computacional quando comparamos as duas tabelas anteriores. Todavia, o tempo computacional na tabela representa apenas o valor médio de quando o AG encontrou a melhor solução pela primeira vez considerando as três execuções. O parâmetro $f = 1000$ continua fornecendo valores para custo de troca com uma porcentagem considerável na função objetivo. Quando $f = 10000$, essa participação ultrapassa metade do valor da função objetivo.

Considerando todos os resultados obtidos até agora, avaliamos que $f = 1000$ estabelece custos de troca mais apropriados ao problema que tentamos simular nas instâncias geradas. Por isso, esse será o valor adotado em todos os testes computacionais realizados nos próximos capítulos.

			L=5		
J	T	CPU(seg.)	$\%s$	$\%\overline{s}$	$\%s\overline{s}$
10	4	871,05	23,08	38,28	61,36
15	4	940,58	27,18	37,18	64,36
10	8	1187,39	20,86	47,85	68,71
10	12	1179,06	21,18	40,78	61,96

Tab. 4.14: $L = 5$, $f = 10000$

4.5 Conclusão

A forma como as instâncias de teste do PCDLPP são geradas foi apresentada neste capítulo. As fórmulas estabelecidas demonstram a dificuldade de se gerar instâncias em um problema envolvendo dois níveis de produção interdependentes. As fórmulas auxiliam mas não são suficientes para garantir que instâncias factíveis serão geradas com a capacidade disponível. Na prática, a determinação das instâncias foi uma etapa bastante árdua do trabalho. A maioria das instâncias criadas eram infactível. As fórmulas aqui apresentadas são o resultado final de um método que permitiu amenizar essa dificuldade.

Toda vez que uma linha é parada para realizar uma troca de produto ou toda vez que um tanque precisa ser preparado para um novo reabastecimento há um custo envolvido. Assim, também foram avaliadas duas formas de se estabelecer os custos de troca para o PCDLPP a partir dos tempos de troca. O objetivo era evitar custos de troca arbitrários e irrelevantes ao problema. A avaliação permitiu determinar custos de troca proporcionais ao tempo de troca e com um impacto considerável no valor total da função objetivo. As porcentagens apresentadas demonstram que o uso de $f = 1000$ permite a obtenção de custos de troca a partir do tempo de preparo que não estão subestimados dentro do custo total da função objetivo.

Capítulo 5

Resolução utilizando GAMS/Cplex

5.1 Introdução

Um método de resolução exata do modelo matemático foi a primeira abordagem usada para encontrar soluções do Problema Conjunto de Dimensionamento de Lotes e Programação da Produção (PCDLPP). O modelo do PCDLPP apresentado no capítulo 3 possui uma grande quantidade de parâmetros, variáveis e restrições. Isso nos levou a escolha de uma ferramenta de modelagem matemática que facilitasse sua implementação e resolução. Optou-se pelo pacote computacional GAMS IDE (*Gams Integrated Development Environment*) versão 2.0.10 devido aos seguintes aspectos:

- Apresenta um ambiente amigável de edição que permite fácil monitoramento da compilação e execução do modelo.
- Apresenta uma linguagem de alto nível que permite uma implementação do modelo bastante concisa e similar à sua formulação matemática.
- Independência entre codificação do modelo e instâncias do problema. Essa separação permite aumentar o tamanho do problema sem alterar a complexidade da sua representação.
- Independência entre modelo e ferramenta de resolução. Diferentes ferramentas de resolução poderão ser aplicadas ao mesmo modelo e às suas instâncias.

. Chamamos de ferramenta de resolução os diversos *solvers* existentes no GAMS. A ferramenta Cplex versão 7.0 foi escolhida porque fois desemvolvida para determinar a solução de problemas de elevada dimensão e complexidade de forma rápida e com pouca interferência do usuário. Inúmeras opções são automaticamente ajustadas levando-se em conta o tipo de problema e aspectos relacionados à alocação de memória.

O PCDLPP é um Problema de Programação Inteira Mista (PPIM) e o Cplex resolve PPIM através de um algoritmo do tipo *branch & cut*. Esse algoritmo soluciona uma série de subproblemas de Programação Linear (PL). Mesmo um simples PPIM é capaz de gerar muitos subproblemas do tipo PL, levando o Cplex a demandar um grande esforço computacional e a requerer uma significativa quantidade de memória. Os testes computacionais realizados demonstram que isto de fato ocorreu durante a resolução do PCDLPP. Por isso, o tempo máximo de execução do GAMS/Cplex em todas as instâncias foi ajustado para 1 hora. Todos os resultados apresentados nas próximas seções foram obtidos em um Pentium IV com 2.8 GHz.

5.2 Resultados para instâncias de pequena dimensão

A geração de instâncias de pequena dimensão foi apresentada no capítulo 4. A tabela 5.1 contém o número de variáveis binárias, variáveis contínuas e o número de restrições presentes no modelo correspondente a cada instância dos conjuntos $L/\overline{L}/J/\overline{J}$ quando $T = 1$.

$L/\overline{L}/J/\overline{J}$	Modelo		
	Var. Binárias	Var. Contínuas	Restrições
2 / 2 / 2 / 1	100	263	281
2 / 2 / 3 / 2	136	499	454
2 / 2 / 4 / 2	142	615	512
3 / 2 / 2 / 1	150	452	419
3 / 2 / 3 / 2	204	880	673
3 / 2 / 4 / 2	213	1098	764
4 / 2 / 2 / 1	176	555	498
4 / 2 / 3 / 2	246	1100	821
4 / 2 / 4 / 2	258	1390	924

Tab. 5.1: Valores do modelo matemático quando $T = 1$

A tabela 5.2 apresenta os resultados obtidos usando Gams/Cplex quando $T = 1$.

Cada combinação $L/\overline{L}/J/\overline{J}$ representa um conjunto de 10 instâncias. A segunda coluna lista o número de soluções ótimas encontradas dentre as 10 instâncias testadas. O valor médio (Méd), máximo (Máx), mínimo(Mín) e o desvio padrão (Std) do Gap e CPU também são listados. O Gap é a distância entre a solução retornada pelo método e o melhor limitante inferior obtido. A solução ótima foi encontrada na maioria das instâncias.

O GAMS/Cplex não encontrou dificuldades em $T = 1$ para solucionar os modelos relativos às instâncias geradas. A tabela 5.3 apresenta informações sobre o modelo de cada instância gerada quando $T = 2$.

$L/\overline{L}/J/\overline{J}$	Soluções Ótimas	Gap(%)				CPU(seg.)			
		Méd	Máx	Mín	Std	Méd	Máx	Mín	Std
2 / 2 / 2 / 1	10	0,00	0,00	0,00	0,00	2,08	10,22	0,09	3,14
2 / 2 / 3 / 2	10	0,00	0,00	0,00	0,00	0,96	4,06	0,09	1,25
2 / 2 / 4 / 2	10	0,00	0,00	0,00	0,00	0,77	2,11	0,22	0,61
3 / 2 / 2 / 1	10	0,00	0,00	0,00	0,00	573,32	2323,84	2,53	839,98
3 / 2 / 3 / 2	10	0,00	0,00	0,00	0,00	151,79	513,77	0,33	218,89
3 / 2 / 4 / 2	10	0,00	0,00	0,00	0,00	46,26	240,52	0,67	72,36
4 / 2 / 2 / 1	2	1,13	3,23	0,00	1,11	3068,64	3600	10,73	1203,42
4 / 2 / 3 / 2	8	0,55	3,03	0,00	1,17	989,39	3600	0,78	1474,24
4 / 2 / 4 / 2	7	0,80	3,05	0,00	1,29	1356,72	3600	18,39	1621,32

Tab. 5.2: Conjunto de instâncias com $T = 1$

$L/\overline{L}/J/\overline{J}$	Modelo		
	Var. Binárias	Var. Contínuas	Restrições
2 / 2 / 2 / 1	210	533	575
2 / 2 / 3 / 2	282	1004	919
2 / 2 / 4 / 2	294	1235	1039
3 / 2 / 2 / 1	315	916	862
3 / 2 / 3 / 2	423	1771	1368
3 / 2 / 4 / 2	441	2206	1552
4 / 2 / 2 / 1	367	1122	1013
4 / 2 / 3 / 2	507	2211	1641
4 / 2 / 4 / 2	531	2790	1865

Tab. 5.3: Valores do modelo matemático quando $T = 2$

$L/\overline{L}/J/\overline{J}$	Soluções Ótimas	Gap(%)				CPU(seg.)			
		Méd	Máx	Mín	Std	Méd	Máx	Mín	Std
2/2/2/1	10	0,00	0,00	0,00	0,00	28,27	107,97	1,02	34,30
2/2/3/2	6	1,50	7,42	0,00	2,51	1703,08	3600	17,09	1697,02
2/2/4/2	6	1,02	4,49	0,00	1,54	1545,06	3600	6,31	1775,13
3/2/2/1	1	0,82	1,71	0,00	0,52	3365,86	3600	1258,02	740,62
3/2/3/2	0	3,15	11,75	0,21	3,66	3600	3600	3600	0,0
3/2/4/2	1	2,21	7,59	0,00	2,21	3260,23	3600	201,67	1074,67
4/2/2/1	0	1,23	2,41	0,27	0,78	3600	3600	3600	0,0
4/2/3/2	1	1,07	3,48	0,00	1,01	3254,55	3600	144,70	1092,69
4/2/4/2	0	4,23	17,45	0,62	4,99	3600	3600	3600	0,0

Tab. 5.4: Conjunto de instâncias com $T = 2$

A tabela 5.4 apresenta os resultados obtidos quando $T = 2$.

O GAMS/Cplex começa a ter problemas para determinar a solução ótima. Além disso, cresce

o tempo computacional necessário para encontrar essas soluções. Observe que nos conjuntos onde a melhor solução factível foi retornada, o GAMS/Cplex necessariamente foi executado até atingir o tempo limite de 1 hora. Esse aumento na dificuldade de encontrar a solução ótima foi acompanhado por um aumento no número de variáveis e restrições presentes no modelo (tabela 5.3).

Os resultados obtidos quando $T = 3$ e $T = 4$ estão nas tabelas 5.5 e 5.6. O método não encontra

$L/\overline{L}/J/\overline{J}$	Soluções Ótimas	Gap(%)				CPU(seg.)			
		Méd	Máx	Mín	Std	Méd	Máx	Mín	Std
2 / 2 / 2 / 1	2	1,23	2,69	0,00	0,87	3146,80	3600	1101,94	961,77
2 / 2 / 3 / 2	3	2,37	6,65	0,00	2,34	3076,89	3600	6,33	1193,28
2 / 2 / 4 / 2	3	6,07	20,09	0,00	6,92	2911,83	3600	188,92	1330,87
3 / 2 / 2 / 1	0	1,49	2,89	0,04	0,86	3600	3600	3600	0,0
3 / 2 / 3 / 2	0	5,79	14,81	1,93	4,21	3600	3600	3600	0,0
3 / 2 / 4 / 2	0	4,70	13,22	1,14	3,50	3600	3600	3600	0,0
4 / 2 / 2 / 1	1	1,54	3,69	0,00	1,09	3281,30	3600	412,27	1008,07
4 / 2 / 3 / 2	0	9,41	29,44	0,58	9,19	3600	3600	3600	0,0
4 / 2 / 4 / 2	0	7,23	14,08	1,67	3,98	3600	3600	3600	0,0

Tab. 5.5: Conjunto de instâncias com $T = 3$

$L/\overline{L}/J/\overline{J}$	Soluções Ótimas	Gap(%)				CPU(seg.)			
		Méd	Máx	Mín	Std	Méd	Máx	Mín	Std
2/2/2/1	1	1,50	3,34	0,00	0,99	3377,44	3600	1373,91	703,97
2/2/3/2	1	5,36	11,75	0,00	4,04	3244,53	3600	44,50	1124,38
2/2/4/2	0	4,29	15,84	0,29	5,17	3600	3600	3600	0,0
3/2/2/1	0	1,71	2,92	0,72	0,62	3600	3600	3600	0,0
3/2/3/2	0	8,25	22,32	1,72	6,27	3600	3600	3600	0,0
3/2/4/2	0	8,54	15,75	1,81	4,93	3600	3600	3600	0,0
4/2/2/1	0	1,71	3,12	0,48	0,82	3600	3600	3600	0,0
4/2/3/2	0	8,79	19,21	1,40	5,62	3600	3600	3600	0,0
4/2/4/2	0	11,02	20,78	3,35	6,16	3244,11	3600	3600	0,04

Tab. 5.6: Conjunto de instâncias com T=4

soluções ótimas na grande maioria das instâncias. O crescimento no valor do Gap demonstra que ocorre uma piora na qualidade da melhor solução factível retornada pelo GAMS/Cplex. As tabelas 5.7 e 5.8 apresentam os valores relativos ao modelo matemático para essas instâncias.

A complexidade do modelo proposto associada ao número considerável de variáveis e restrições presentes nos modelos quando $T = 3$ e $T = 4$ justificam a dificuldade encontrada pelo algoritmo *branch & cut* do GAMS/Cplex em determinar soluções ótimas dentro de 1 hora. A figura 5.1 exemplifica a evolução na dimensão dos modelos solucionados pelo GAMS/Cplex no decorrer dos

$L/\overline{L}/J/\overline{J}$	Modelo		
	Var. Binárias	Var. Contínuas	Restrições
2 / 2 / 2 / 1	320	803	863
2 / 2 / 3 / 2	574	2014	1835
2 / 2 / 4 / 2	598	2475	2075
3 / 2 / 2 / 1	480	1380	1303
3 / 2 / 3 / 2	642	2662	2071
3 / 2 / 4 / 2	669	3314	2350
4 / 2 / 2 / 1	558	1689	1536
4 / 2 / 3 / 2	768	3322	2466
4 / 2 / 4 / 2	804	4190	2799

Tab. 5.7: Valores do modelo matemático quando $T = 3$

$L/\overline{L}/J/\overline{J}$	Modelo		
	Var. Binárias	Var. Contínuas	Restrições
2 / 2 / 2 / 1	430	1073	1163
2 / 2 / 3 / 2	574	2014	1827
2 / 2 / 4 / 2	598	2475	2075
3 / 2 / 2 / 1	645	1844	1752
3 / 2 / 3 / 2	861	3553	2764
3 / 2 / 4 / 2	897	4422	3140
4 / 2 / 2 / 1	749	2256	2039
4 / 2 / 3 / 2	1029	4433	3335
4 / 2 / 4 / 2	1077	5590	3735

Tab. 5.8: Valores do modelo matemático quando $T = 4$

macro-períodos através do conjunto de instâncias 4/2/4/2.

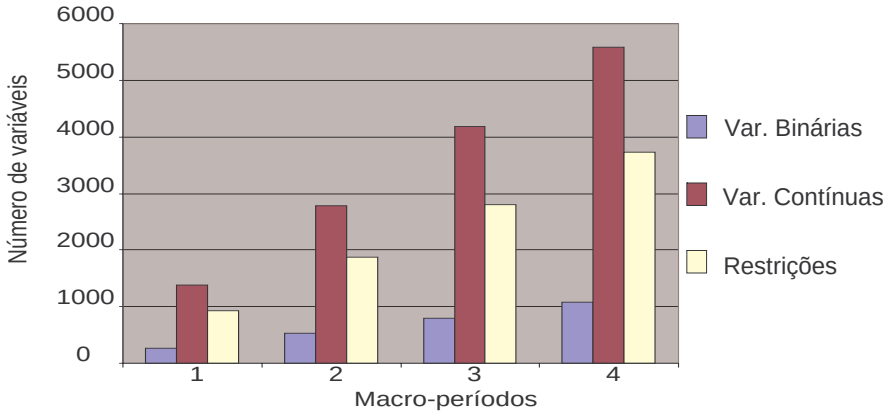

Fig. 5.1: Evolução do número de variáveis e restrições em 4/2/4/2.

5.3 Resultados para instâncias de elevada dimensão

Os resultados computacionais anteriores indicam a inviabilidade da solução exata de instâncias de elevada dimensão dentro do tempo limite estabelecido. Um aumento no tempo de execução talvez permitisse obter soluções factíveis. Porém, os valores do Gap na tabela 5.6 demonstram a queda na qualidade de tais soluções.

Acreditamos que 1 hora utilizando uma ferramenta robusta como o GAMS/Cplex já é um tempo limite considerável. Aumentar esse tempo não seria razoável, se pretendemos aqui trabalhar com metodologias que possam vir a ser utilizadas em situações reais. Ao invés de aprofundar a análise de resultados dispendiosos em termos de tempo computacional, optamos por desenvolver alternativas que possibilitassem encontrar soluções dentro do tempo limite de 1 hora. Nesse caso, a única abordagem com GAMS/Cplex que mostrou-se viável foi através da resolução de versões relaxadas do modelo matemático. Três versões serão avaliadas:

Relax1: Todas as variáveis binárias do modelo matemático são relaxadas. Trata-se de uma relaxação linear do modelo.

Relax2: As variáveis u_{ls} e $\bar{u}_{ks}$ permanecem binárias e as demais são relaxadas.

Relax3: As variáveis u_{ls}, $\bar{u}_{ks}$, x_{jls} e $\bar{x}_{jks}$ permanecem binárias e as demais são relaxadas.

Gupta and Magnusson (2005) realizaram esse tipo de análise ao solucionar um Problema de Dimensionamento de Lote Capacitado (PDLC) com custos e tempos de troca dependentes da seqüência. O PDLC proposto apresenta uma formulação complexa que também dificulta a resolução de instâncias de elevada dimensão. Os autores decidiram então determinar limitantes inferiores para essas instâncias, resolvendo versões do modelo com algumas variáveis binárias relaxadas. Diferentes combinações de relaxação de variáveis binárias foram avaliadas e essas versões relaxadas do PDLC foram solucionadas usando o *solver* Cplex.

As três versões relaxadas aqui propostas também foram solucionadas usando Cplex (GAMS/Cplex). A tabela 5.9 contém o número de soluções ótimas (Ot), o número de soluções inteiras factíveis mas não ótimas (Fac) e o número de instâncias onde soluções inteiras não foram retornadas (N Sol.) dentro do tempo limite de 1 hora.

Relax1 retorna soluções ótimas para todas as instâncias com $J = 10$, oito soluções ótimas e duas factíveis com $J = 15$. Soluções ótimas já passam a ser mais difíceis de se determinar quando variáveis binárias estão presentes em Relax2. O número de soluções factíveis retornadas é maior e em alguns casos nenhuma solução é retornada. A situação é ainda pior em Relax3 quando apenas em um conjunto de instânias (10/5/4) cinco soluções factíveis foram obtidas. Os tempos computacionais estão na tabela 5.10.

$J/\overline{J}/T$	Relax1			Relax2			Relax3		
	Ot.	Fac	N sol.	Ot.	Fac	N sol.	Ot.	Fac	N sol.
10/5/4	10	0	0	6	4	0	0	5	5
15/8/4	8	2	0	0	10	0	0	0	10
10/5/8	10	0	0	0	8	2	0	0	10
15/8/8	8	2	0	0	2	8	0	0	10
10/5/12	10	0	0	0	10	0	0	0	10
15/8/12	8	2	0	0	0	10	0	0	10

Tab. 5.9: Soluções obtidas para $L = \overline{L} = 5$

$J/\overline{J}/T$	Relax1				Relax2			
	Avg	Max	Min	Std	Avg	Max	Min	Std
10/5/4	19,63	96,72	3,05	28,21	1876,71	3600,00	239,45	1519,71
15/8/4	1343,70	3600,00	162,69	1483,76	3600,00	3600,00	3600,00	0,00
10/5/8	247,49	1295,91	22,42	377,28	3600,00	3600,00	3600,00	0,00
15/8/8	1275,38	3600,00	184,00	1331,37	3600,00	3600,00	3600,00	0,00
10/5/12	48,47	158,67	8,97	46,48	3600,00	3600,00	3600,00	0,00
15/8/12	929,51	2690,31	92,48	964,67	3600,00	3600,00	3600,00	0,00

Tab. 5.10: Tempo computacional de Relax1 e Relax2

A tabela 5.11 apresenta o número de variáveis e restrições envolvidas na solução de modelos exatos. A tabela 5.12 apresenta uma comparação do número de variáveis entre as versões relaxadas

$L/\overline{L}/J/\overline{J}/T$	Modelo		
	Var. Binárias	Var. Contínuas	Restrições
5/ 5 / 10 / 5 / 4	4810	71961	23857
5 / 6 / 15 / 8 / 4	8724	208172	65166
5/ 5 / 10 / 5 / 8	9670	143961	47549
5 / 6 / 15 / 8 / 8	17508	416388	132258
5/ 5 / 10 / 5 / 12	14530	215961	70657
5 / 6 / 15 / 8 / 12	26292	624604	197318

Tab. 5.11: Número de variáveis e restrições - Modelo Exato

do modelo. Um número elevado de variáveis e restrições está presente no modelo exato (tabela 5.11), deixando evidente a dificuldade de se solucionar através de um método exato um modelo complexo e com essa dimensão de variáveis e restrições. O número de variáveis binárias diminui nas versões relaxadas. Porém, o número de variáveis binárias em Relax3 fica acima de 1000 em todas as instâncias. Relax2 também apresenta uma quantidade relevante de variáveis binárias.

$L/\overline{L}/J/\overline{J}/T$	Número de Variáveis		
	Total Var.	Var. Binárias-Relax2	Var. Binárias-Relax3
5/5/10/5/4	76771	160	1460
5/5/15/8/4	216896	240	3280
5/5/10/5/8	153631	320	2920
5/5/15/8/8	433896	480	6560
5/5/10/5/12	230491	480	4380
5/5/15/8/12	650896	720	9840

Tab. 5.12: Estatísticas dos Modelos Matemáticos

5.4 Conclusão

Os resultados computacionais obtidos nas instâncias de pequena dimensão demonstraram que mudanças suaves no número de macro-períodos, produtos e linhas são suficientes para impedir a obtenção de soluções ótimas dentro do tempo limite de 1 hora.

Enquanto as soluções ótimas se tornam inviáveis nas instâncias de pequena dimensão, factibilidade passa a ser o problema em instâncias de elevada dimensão. Obter pelo menos uma solução factível, ainda que para versões relaxadas do modelo, torna-se um problema. Nosso desejo inicial de obter limitantes inferiores usando versões relaxadas do modelo acabou sendo inviável. Por exemplo, Relax2 ainda conseguiu obter soluções ótimas mas retornou apenas soluções factíveis na grande maioria das instâncias. Dessa forma, não podemos fazer a mesma análise de Gupta and Magnusson (2005) já que esse limitante inferior não é o melhor possível.

Os resultados obtidos para instâncias de elevada dimensão servem para ilustrar a dificuldade de se obter sequer soluções factíveis nesse tipo de problema. Eles também demonstram que o modelo não é adequado para gerar limitantes. Assim, o modelo não nos enconraja a usar procedimentos de decomposição e relaxação, nem procedimentos do tipo *fix-and-relax*, para tentar obter boas soluções viáveis a partir do modelo original. Apresentamos no apêndice alguns resultados obtidos numa tentativa de determinar soluções usando um procedimento baseado em horizonte rolante e *fix-and-relax*.

Por isso, o próximo passo é o desenvolvimento de meta-heurísticas que possam encontrar soluções factíveis de boa qualidade dentro do tempo limite desejado. Os resultados obtidos para as instâncias de pequena dimensão é que irão servir como critério de avaliação das meta-heurísticas.

Capítulo 6

Resolução usando Meta-heurísticas

6.1 Introdução

Os resultados computacionais obtidos usando a ferramenta GAMS/Cplex demonstraram que encontrar a solução ótima ou mesmo factível nem sempre era possível dentro de um tempo computacional razoável. Assim, optamos por solucionar o problema por meio de uma abordagem baseada em meta-heurísticas. Um procedimento heurístico em geral encontra rapidamente uma solução para um problema mas sem garantias de que ela seja ótima. Uma meta-heurística se diferencia de uma heurística por adicionar mais inteligência ao processo de busca por soluções. Tal metodologia procura escapar de ótimos locais e percorrer áreas mais amplas dentro do espaço de soluções possíveis. Busca Tabu (Glover (1989)), Simulated Annealing (Kirkpatrick et al. (1983)) e Algoritmos Genéticos (Holland (1975)) são exemplos de meta-heurísticas consagradas por bons resultados obtidos para uma vasta gama de problemas.

Será descrito neste capítulo os métodos baseados em Algoritmos Genéticos (AG) e Algoritmos Meméticos (AM) (seções6.2 e 6.3). Também será apresentado os resultados computacionais obtidos. O ambiente de otimização Np-Opt (Mendes (2003)) foi utilizado na implementação de tais métodos. NP-Opt segue a filosofia da programação orientada a objeto reduzindo o trabalho de implementação através da reutilização de código. O ambiente apresenta uma série de classes implementadas em linguagem de programação Java que permite uma rápida e fácil inserção de novos códigos. O usuário faz uso das classes existentes nas implementações gerais e cria códigos apenas para aspectos específicos ao seu método. O ambiente divide-se em quatro partes:

- Interface com usuário inteiramente gráfica permitindo fácil acesso aos recursos disponíveis no ambiente.

- Classes e métodos implementados para instâncias, indivíduos, operadores genéticos, buscas

locais, entre outros.

- Processamento distribuído com rotinas que realizam a alocação de tarefas.
- Interface de saída com os resultados fornecidos pelos métodos.

Atualmente o ambiente NP-Opt possui cinco problemas implementados para testes e estudos dos usuários: Sequenciamento em Máquina Simples, Máquinas Paralelas e *Flowshop*, *Gate Matrix Layout* e Ordenamento de Genes. Maiores detalhes sobre o ambiente NP-Opt podem ser encontrados em Mendes (2003).

6.2 Algoritmo Genético

6.2.1 Aspectos gerais

Algoritmo Genético (AG) designa um método computacional que simula processos biológicos de recombinação genética, mutação e seleção gerando soluções para um dado problema. O primeiro trabalho sobre AG foi apresentado por Holland (1975) e detalhes sobre AG podem ser encontrados em Goldberg (1989), Davis (1991), Michalewicz (1996), Bäck et al. (2000a) e Bäck et al. (2000b). A figura 6.1 apresenta uma idéia geral do AG.

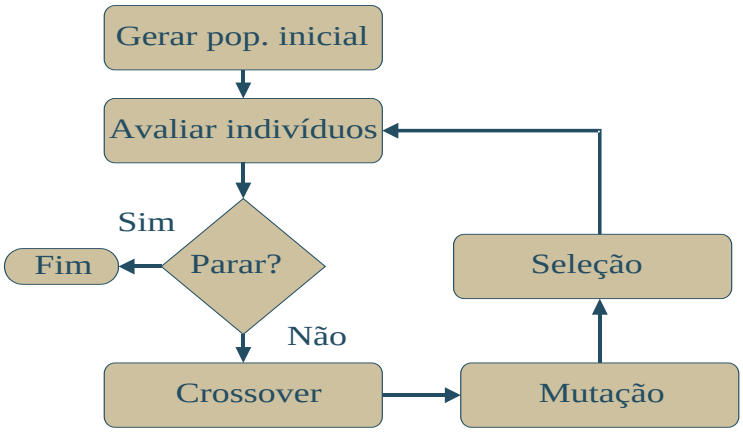

Fig. 6.1: Algoritmo Genético - Fluxograma.

Diversos mecanismos que permitem guiar uma busca por soluções são adaptados do contexto da teoria da evolução dos seres vivos. Assim, indivíduo ou cromossomo no contexto do AG é a representação de uma possível solução do problema matemático associado. As informações que fornecem a solução estão de alguma forma codificadas no indivíduo, onde os genes representam cada

atributo (ou informação) codificada nesse indivíduo. O conjunto de indivíduos forma uma população e um AG pode ter uma ou várias populações.

O primeiro passo no AG é responsável por gerar a população inicial sobre a qual o processo evolutivo será executado. Seguindo a inspiração biológica, o AG avalia cada indivíduo de forma a identificar aqueles mais aptos a permanecerem na população. A aptidão de um indivíduo é determinda por uma função chamada *fitness*. O *fitness* procura indicar quais indivíduos representam as melhores soluções para o problema em questão. Criada uma população e avaliado cada indivíduo, o processo evolutivo é efetuado a cada geração (iteração do algoritmo) por operadores genéticos de seleção, recombinação (*crossover*) e mutação.

A seleção determina quais indivíduos irão reproduzir e quais irão permanecer na população. Os indivíduos com melhor valor de *fitness* são em geral selecionados, mas isto dependerá do tipo de seleção adotada pelo AG implementado. A recombinação procura efetuar trocas de informações codificadas nos indivíduos para que novas e, se possível, melhores soluções sejam criadas. A pressão seletiva pode ocasionar uma rápida homogenização da população, o que leva a uma prematura convergência do algoritmo. O operador de mutação procura evitar isso alterando a informação codificada em alguns indivíduos para restaurar uma saudável heterogenização da população. O procedimento termina quando um número adequado de gerações foi atingido.

O AG deste trabalho segue a estrutura já implementada no ambiente NP-Opt. O pseudocódigo abaixo foi retirado de Mendes (2003) e descreve em linhas gerais o tipo de AG presente no NP-Opt.

```
Method algGeneticoMultiPopulacional;
begin
    repeat
        for i = 1 to numberOfPopulations do
            inicializaPopulação(pop(i ));
            avaliaFitnessPopulação(pop(i ));
            estruturaPopulação(pop(i ));
            repeat
                for j = 1 to numberOfRecombinations do
                    selecionaPais(individuoA,individuoB)⊆ pop(i);
                    novoInd = recombina(individuoA, individuoB);
                    if (efetuaMutação novoInd) then novoInd = mutação(novoInd);
                    avaliaFitnessIndivíduo(novoInd);
```

```
                    inserePopulação(novoInd, pop(i ));
                end
                estruturaPopulação(pop(i ));
            until(populaçãoConvergiu pop(i ));
        end
        for i = 1 to numberOfPopulations do
            efetuaMigração pop(i );
        end
    until(condição de parada)
end
```

O algoritmo anterior apresenta também os diversos métodos disponíveis no NP-Opt. Optamos por utilizar nesse trabalho um algoritmo genético multi-populacional, onde várias populações evoluem no decorrer das gerações. A abordagem multi-populacional no AG permite maior exploração do espaço de busca já que indivíduos em populações diferentes seguem em geral caminhos evolutivos distintos. Esse é o conceito de *genetic drift* apresentado em Weiner (1995). Um objetivo desse trabalho foi determinar soluções ao menos factíveis para instâncias de elevada dimensão em um tempo computacional razoável. Logo, a abordagem multi-populacional atende às nossas expectativas ao prestigiar o aspecto exploratório na busca de soluções. Uma abordagem uni-populacional intensifica a busca num espaço mais limitado e isso diminui a possibilidade de se encontrar soluções de melhor qualidade ou mesmo soluções factíveis.

O método **estruturaPopulação**(pop(i) organiza os indivíduos dentro de cada população. Os operadores genéticos de seleção (**selecionaPais**(individuoA,individuoB)), recombinação (**recombina** (individuoA, individuoB)) e mutação (**mutação**(novoInd)) são executados enquanto a população i avaliada não convergir. Uma população converge, se após um certo número de iterações, nenhum novo indivíduo é inserido. Se todas as populações avaliadas convergiram, antes do critério de parada ser satisfeito, ocorrerá uma migração entre as populações (**efetuaMigração** pop(i)) e uma nova inicialização das populações. Todavia, essa nova inicialização poupa os melhores indivíduos e aqueles indivíduos que acabaram de migrar. Trata-se de uma reinicialização parcial da população.

6.2.2 Indivíduos e Codificação

Os indivíduos ou cromossomos representam possíveis soluções para o problema. Os diferentes atributos que caracterizam cada indivíduo são chamados genes e as informações que descrevem uma solução do problema estão codificadas nos genes de cada indivíduo. A idéia adotada neste trabalho

foi estabelecer uma codificação que permitisse ao indivíduo armazenar informações suficientes para determinar a programação e o dimensionamento de produtos e xaropes nas linhas e tanques. Uma representação compacta como aquelas baseadas em valores binários, freqüentemente utilizadas em AG (Goldberg (1989)), dificilmente conseguiria representar uma solução para o PCDLPP. Optamos por utilizar uma representação mais complexa, conforme pode ser visto na figura 6.2.

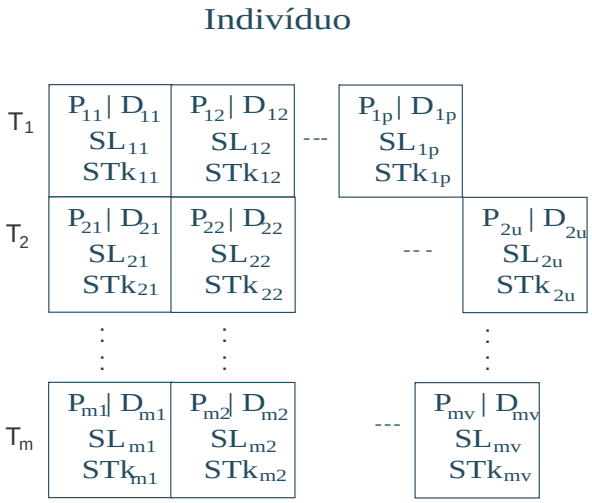

Fig. 6.2: Indivíduo.

Um indivíduo será representado por uma lista de macro-períodos $T_1, T_2, ..., T_m$ com diferentes quantidades de genes em cada período. Por exemplo, temos p genes no macro-período T_1, u genes em T_2 e v genes no último macro-período na figura 6.2. As seguintes informações estão contidas em um gene:

- P_{mn}: número do n-ésimo produto a ser produzido no macro-período m.
- D_{mn}: tamanho do lote do produto P_{mn}.
- SL_{mn}: seqüência de possíveis linhas onde D_{mn} pode ser produzido.
- STk_{mn}: seqüência de possíveis tanques, onde o xarope de D_{mn} pode ser armazenado.

A demanda d_{it} de um produto i no macro-período t pode ser dividida em lotes D_{mn} distribuídos pelos genes contidos nos macro-períodos $t, t-1, t-2, ..., 1$. Uma seqüência de linhas SL_{mn} onde o lote D_{mn} pode ser produzido é definida da seguinte forma:

$$SL_{mn} = (\alpha_1, ..., \alpha_k)$$

onde,

- k é o número de linhas presentes na seqüência.
- α_i contém o número da linha na i-ésima posição da seqüência. $\alpha_i \in \{1, ..., L\}$, L é o número de linhas.

Uma seqüência de tanques é definida da seguinte forma:

$$STk_{mv} = (\beta_1, ..., \beta_k),$$

onde,

- k é o número de tanques presentes na seqüência.
- β_i é o número que indica onde e como o xarope do produto será armazenado. $\beta_i \in \{1, 2, ..., 2\overline{L}\}$, $\overline{L}$ é o número de tanques.

O valor de α_i indica apenas a linha onde o produto será produzido enquanto β_i indica não só o tanque onde o xarope será armazenado, mas como ele será armazenado. O tanque j onde o xarope será armazenado é obtido a partir de β_i da seguinte forma:

$$j = \begin{cases} \beta_i, & 1 \leq \beta_i \leq \overline{L}; \\ \beta_i - \overline{L}, & \overline{L} < \beta_i \leq 2\overline{L}. \end{cases}$$

Considere o lote do xarope u relativo à produção de D_{mn} unidades do produto P_{mn}. A decisão de como esse lote do xarope u será atribuído ao tanque j é tomada de acordo com os seguintes critérios:

1. Se o tanque j estiver ocupado, ainda que parcialmente, por um xarope v com $u \neq v$, o xarope u irá pertencer a uma nova atribuição de lote ao tanque. Dois xaropes diferentes não podem ser armazenados simultaneamente em j.
2. Se o tanque j estiver completamente ocupado, o xarope u irá pertencer a uma nova atribuição de lote ao tanque.
3. Se o tanque j estiver vazio, ele será imediatamente ocupado por u.
4. Se o tanque j já estiver ocupado pelo xarope u e a ocupação mínima do tanque ainda não foi satisfeita, o lote do xarope u será atribuido a j.
5. Se $1 \leq \beta_i \leq \overline{L}$, o xarope u irá fazer parte de uma nova atribuição de lote ao tanque $j = \beta_i$, desde que as condições anteriores sejam respeitadas.
6. Se $\overline{L} < \beta_i \leq 2\overline{L}$, o xarope u irá fazer parte do lote já atribuído ao tanque $j = \beta_i - \overline{L}$, desde que as condições anteriores sejam respeitadas.

O principal objetivo dos critérios (5) e (6) é permitir que um tanque possa ser ocupado parcialmente. Os demais critérios estabelecidos têm prioridade sobre os dois últimos porque procuram satisfazer as restrições de ocupação dos lotes e capacidade dos tanques (veja restrições (3.30)-(3.34) no capítulo 3). A figura 6.3 ilustra dois lotes de um tanque, onde primeiro ocorre ocupação total e depois ocupação parcial. A linha pontilhada representa a capacidade mínima do tanque.

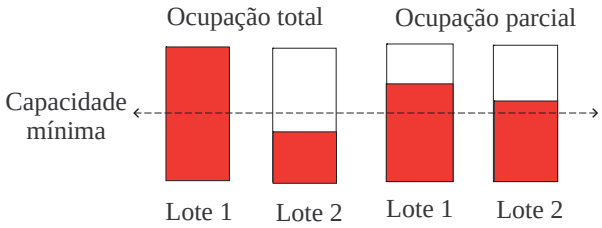

Fig. 6.3: Exemplo de indivíduos.

Se considerarmos apenas a inserção do lote de um tanque até que toda capacidade disponível seja ocupada, o xarope remanescente poderá não ser suficiente para atingir a capacidade mínima do tanque no próximo lote. Uma alternativa seria preencher o tanque até atingir o nível mínimo, mas isso poderia levar a um excesso na produção de itens que utilizam tal xarope. Se permitirmos uma ocupação parcial do tanque, essa situação poderá ser evitada. Todavia, caso essa situação não possa ser evitada, o tanque será preenchido até se atingir a capacidade mínima e o excesso pode ser usado na produção de produtos que utilizam esse xarope. Isso aumenta o custo de estoque e produção dos produtos. Caso não exista capacidade disponível na linha para produzir o xarope em excesso, o mesmo será descartado. Isso ocasiona um aumento no custo do xarope e poderá ocasionar um aumento no custo de estoque, pois o xarope em excesso pode permanecer no tanque durante vários períodos até ser descartado.

Dorndorf and Pesch (1995) também utilizam uma representação complexa para indivíduos, onde os genes armazenam leis para selecionar tarefas a serem programadas em um problema de *job shop scheduling*. Nossa codificação está mais próxima de Kimms (1999) que utiliza regras de atribuição em um problema multi-nível de dimensionamento e programação de lotes em máquinas paralelas. Nesse trabalho, um indivíduo é uma matriz com M colunas e T linhas. Cada entrada da matriz é uma regra (gene) usada para selecionar itens a serem produzidos em determinada linha $m \in M$ durante um macro-período $t \in T$.

Os indivíduos de uma população são aqui gerados de acordo com o seguinte algoritmo:

Passo1: Para cada macro-período $t = 1, ..., T$ faça

Passo1.1: Repita

Passo1.1.1: Selecione aleatoriamente um produto $i \in \{1, 2, ..., J\}$ com $d_{it} > 0$.

Passo1.1.2: Ajuste $maxDem = d_{it}$.

Passo1.1.3: Enquanto $maxDem > 0$, faça

Passo1.1.3.1: Selecione aleatoriamente $T_m \in \{1, 2, ..., t\}$

Passo1.1.3.2: Determine aleatoriamente $D_m \neq 0$ com $D_m \in [0, maxDem]$

Passo1.1.3.3: Insira D_m na primeira posição de T_m.

Passo1.1.3.4: Ajuste $maxDem = maxDem - D_m$.

Passo1.1.3.5: Para $i = 1, ..., k$;

Gerar aleatoriamente valores para STk_{mn} e SL_{mn} com $\alpha_i \in \{1, ..., L\}$ e $\beta_i \in \{1, ..., 2\overline{L}\}$.

Passo1.2 Até que toda demanda em t tenha sido distribuída no indivíduo

Passo2: Fim.

Um pequeno exemplo nos ajudará a visualizar o tipo de codificação aqui apresentada. Suponha que existam duas linhas e dois tanques disponíveis para produção e armazenagem de dois produtos e seus respectivos xaropes. As demandas desses produtos devem ser atendidas de acordo com a tabela 6.1 abaixo. A figura 6.4 apresenta duas possíveis codificações.

	T_1	T_2
P_1	100	100
P_2	100	100

Tab. 6.1: Produtos e demandas

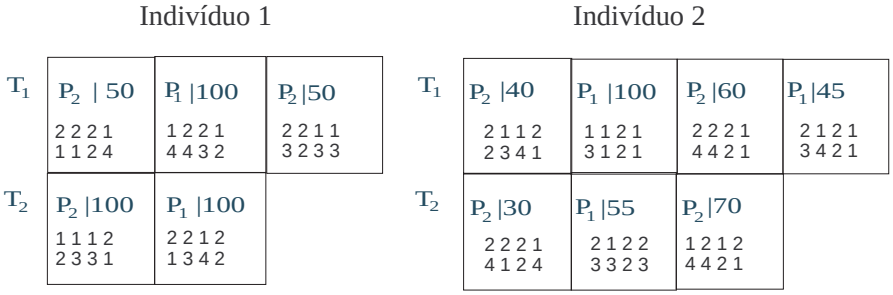

Fig. 6.4: Exemplo de indivíduos.

As demandas dos produtos estão distribuidas dentro dos respectivos macro-períodos no indivíduo 1. A demanda de P_2 em T_1 foi dividida entre dois genes. No indivíduo 2, parte da demanda de P_1 em T_2 é produzida em T_1. Observe também que a seqüência de linhas e tanques poderá conter valores

repetidos de $\alpha_i \in \{1,2\}$ e $\beta_i \in \{1,2,3,4\}$ com $L = \overline{L} = 2$ e $k = 4$. Após gerar os indivíduos da população inicial, o próximo passo é determinar a decodificação da solução representada pelo indivíduo e sua avaliação através da função de *fitness*.

6.2.3 Decodificação e função de *fitness*

A programação e o dimensionamento de lotes nas linhas e tanques são estabelecidos a partir das informações codificadas nos genes de um indivíduo. Os genes de cada macro-período são decodificados seguindo o fluxograma da figura 6.5. O método inicia a alocação das demandas de cada indivíduo

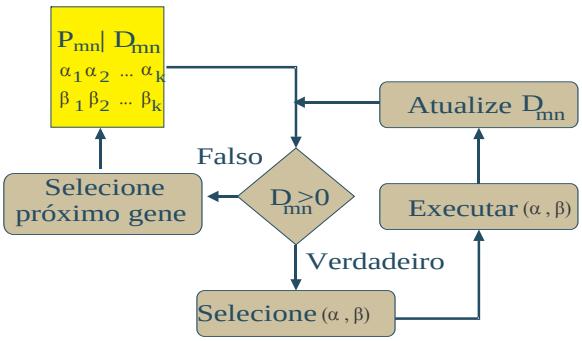

Fig. 6.5: Decodificação - Fluxograma.

a partir das informações contidas nos genes pertencentes ao último macro-período. A idéia é evitar infactibilidades ao se estabelecer a programação do fim para o início dentro de cada macro-período. A codificação estabelece em qual macro-período cada lote será produzido, mas o tempo de preparo dos lotes nos tanques pode ocorrer em um macro-período anterior. Ao estabelecer a programação a partir do final de um macro-período, podemos deslocar os tempos de preparo dos tanques para que ocorram em macro-períodos anteriores. Apesar disso, é possível que a decodificação dos genes de um indivíviduo leve ao não atendimento de demandas no decorrer do processo.

Para cada gene o algoritmo de decodificação tenta programar o lote na posição n do macro-período m, D_{mn}, utilizando a seqüência de linhas e tanques disponíveis no gene. Enquanto $D_{mn} > 0$, um par (α_i, β_i) será selecionado pela ordem em que aparece nas respectivas seqüências. Por isso, o número de linhas e tanques em SL_{mn} e STk_{mn} é sempre o mesmo. Se houver capacidade disponível na linha e no tanque retornados pelo par (α_i, β_i), D_{mn} ou parte de D_{mn} é programado. Caso a capacidade disponível permita que apenas parte de D_{mn} seja programado, o próximo par $(\alpha_{i+1}, \beta_{i+1})$ será selecionado para alocar o valor remanescente.

Vamos ilustrar o algoritmo de decodificação através de um exemplo. Considere os dados da tabela 6.1 do exemplo anterior e a codificação do indivíduo 2 reapresentada na figura 6.6 abaixo.

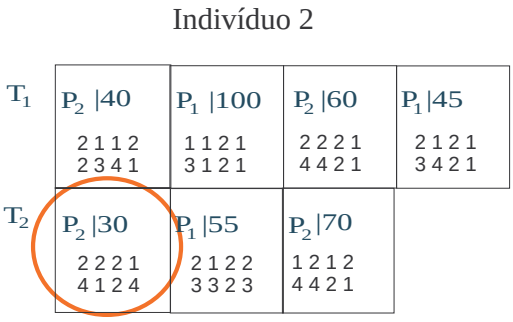

Fig. 6.6: Indivíduo 2.

O processo começa pelo primeiro gene localizado no último macro-período, esse gene aparece circulado na figura 6.6. Um lote de 30 unidades do produto P2 ($D_{21} = 30$) deve ser produzido. O primeiro par $(\alpha_1, \beta_1) = (2, 4)$ é utilizado na determinação da linha e do tanque onde o lote será programado. A figura 6.7 ilustra uma possível decodificação desse gene. A decodificação insere cada

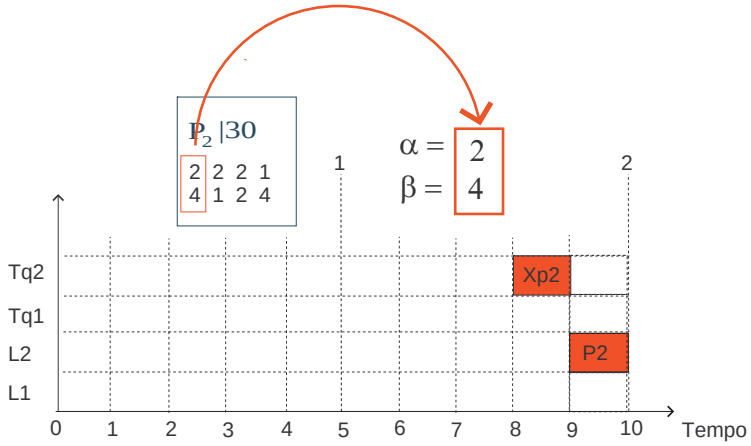

Fig. 6.7: Decodificação do primeiro gene em T_2.

produto a partir do final do horizonte de tempo considerado. Nesse caso, a decodificação programa produtos a partir do final do macro-período T_2. P_2 deve ser produzido na linha 2 já que $\alpha_1 = 2$. O tempo de processamento do produto em L_1 aparece na figura 6.7 ocupando o décimo micro-período. O xarope de P_2 é Xp_2 e deverá ser armazenado no tanque $j = \beta_1 - \overline{L}$ já que $\overline{L} \leq \beta_1 \leq 2\overline{L}$. Assim, $j = 4 - 2 = 2$ com $\overline{L} = 2$. Como o tanque está vazio, Xp_2 será inserido imediatamente no tanque $j = 2$ de acordo com os critérios de atribuição estabelecidos na seção anterior. O tempo gasto no preparo do tanque está na figura 6.7 e iremos assumir que será sempre de 1 micro-período para os dois xaropes aqui considerados. Observe que esse tempo de preparo foi ajustado para terminar no micro-período anterior ao início da produção de P_2 em L_1. Isso está de acordo com a restrição (3.47)

do modelo matemático, onde um produto deve ser produzido apenas em micro-períodos seguintes ao término do tempo de preparo do respectivo xarope.

O par $(\alpha_1, \beta_1) = (2, 4)$ permitiu a inserção de todo o lote $D_{21} = 30$ de P_2, logo os outros pares (α, β) não são utilizados. O próximo passo é inserir o lote do gene seguinte em T_2(figura 6.8).

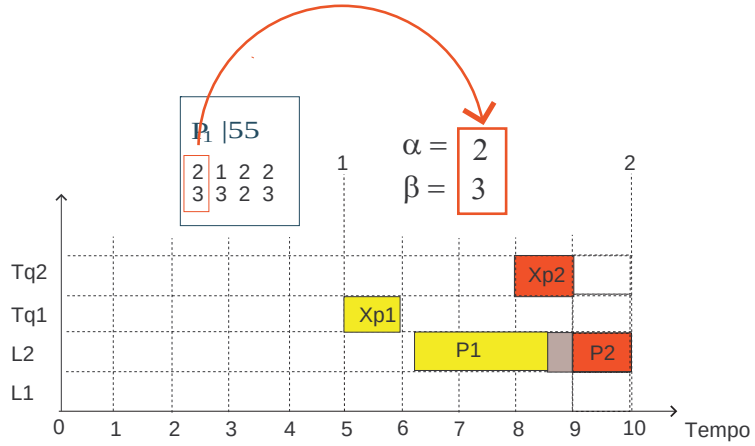

Fig. 6.8: Decodificação do segundo gene em T_2.

O lote com 55 unidades do produto P_1 será produzido em L_2 ($\alpha_1 = 2$) ocasionando um tempo de preparo na linha já que P_2 é produzido em seguida. O xarope de P_1 (Xp_1) é armazenado imediatamente em $j = 1$, pois o tanque está vazio. O tempo de preparo de Xp_1 também aparece na figura 6.8 exatamente no micro-período anterior ao início da produção de P_1 em L_2.

A figura 6.9 apresenta a decodificação do terceiro gene presente em T_2. O restante da demanda

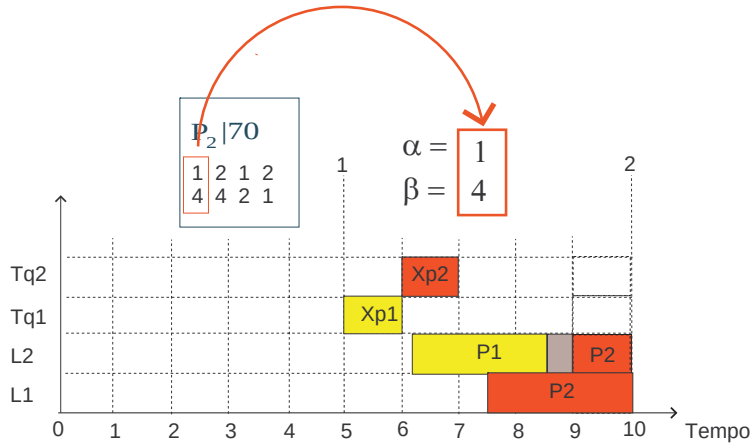

Fig. 6.9: Decodificação do terceiro gene em T_2.

de P_2 em T_2 será produzida em L_1 já que $\alpha = 1$. O valor de $\beta_1 = 4$ indica que o xarope deverá fazer

parte do lote de Xp_2 já armazenado no tanque $j = 4 - 2 = 2$. Isso irá provocar uma antecipação no tempo de preparo do tanque para que termine no micro-período anterior ao início da produção de P_2 em L_1. Assim, todo o lote de Xp_2 armazenado no tanque estará pronto para atender a produção de P_2 nas linhas L_1 e L_2 a partir do final do sétimo micro-período. O processo de decodificação continua nos genes alocados em T_1. A figura 6.10 apresenta a decodificação do primeiro gene em T_1.

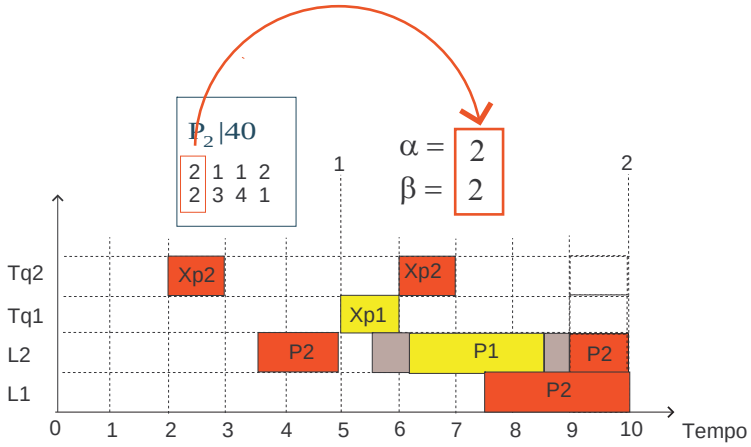

Fig. 6.10: Decodificação do primeiro gene em T_1.

O lote $D_{11} = 40$ de P_2 será produzido em L_2 já que $\alpha_1 = 2$. O valor de $\beta_1 = 2$ indica que o xarope deverá fazer parte de um novo lote do tanque $j = 2$. Vamos supor que a capacidade mínima do tanque já tenha sido atingida, portanto um novo lote será programado no tanque e seu tempo de preparo deverá ocorrer no micro-período anterior ao início da produção de P_2.

A figura 6.11 apresenta a decodificação do próximo gene em T_1. A linha $\alpha_1 = 1$ é utilizada para

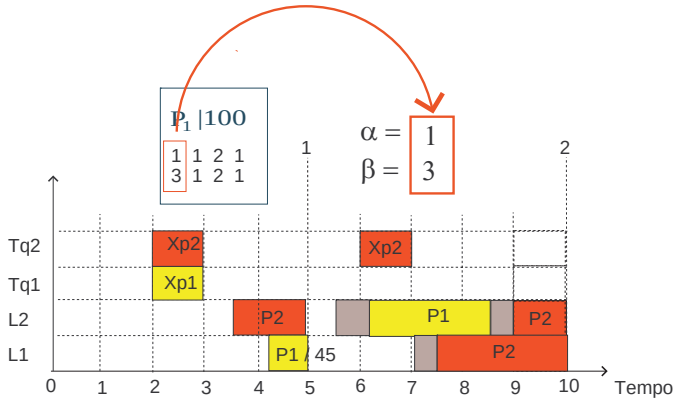

Fig. 6.11: Decodificação do segundo gene em T_1.

produzir $D_{12} = 100$ unidades de P_1. Além disso, $\beta_1 = 3$ indica que o Xp_1 deverá ser armazendo no

lote já programado no tanque $j = 4 - 3 = 1$. Vamos supor que a capacidade disponível no tanque permitiu a inserção de xarope suficiente para produzir apenas 45 unidades de P_1. O tempo de preparo de Xp_1 foi antecipado para o terceiro micro-período e, uma vez que todo o lote não foi produzido, o próximo par (α_2, β_2) é selecionado para programar o lote remanescente $D_{12} = 100 - 45 = 55$. A programação do lote remanescente é apresentada na figura 6.12.

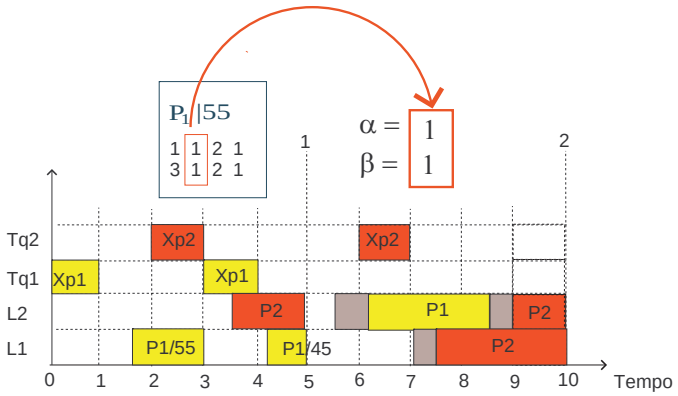

Fig. 6.12: Decodificação final do segundo gene em T_1.

A demanda remanescente foi produzida na linha $\alpha_2 = 1$ e um novo lote de Xp_1 foi ajustado no tanque $j = 1$ já que $\beta_2 = 1$. O próximo gene de T_1 passa a ser decodificado (figura 6.13).

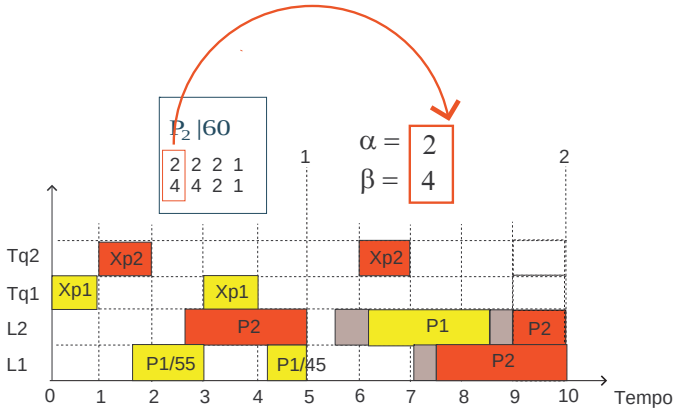

Fig. 6.13: Decodificação do terceiro gene em T_1.

A mesma linha $\alpha = 2$ é selecionada para produzir P_2 e não há tempo de preparo em L_2 já que se trata do mesmo produto. O valor $\beta = 4$ indica que Xp_2 deve integrar o lote já armazenado no tanque $j = 4 - 2 = 2$. Isso provoca uma antecipação no tempo de preparo de Xp_2 que passa para o segundo micro-período. A decodificação do último gene em T_1 segue na figura 6.14.

O valor $\beta = 3$ leva à escolha do tanque $j = 3 - 2 = 1$, onde Xp_2 irá integrar o lote arrmazenado. P_1 será programado em L_2 ($\alpha = 2$) acarretando um tempo de troca já que P_2 é produzido em seguida.

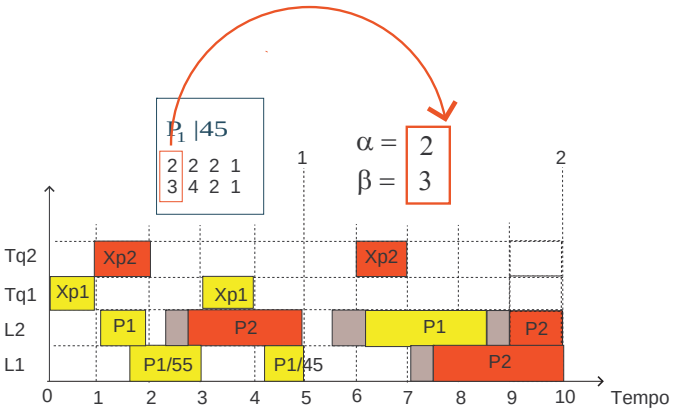

Fig. 6.14: Decodificação do quarto gene em T_1.

Observe que o tempo de troca é curto o suficiente para que a produção de P_1 fosse concluída no terceiro micro-período. Todavia, pela hipótese do modelo matemático, dois produtos não podem ser produzidos dentro de um mesmo micro-período (veja restrições (3.3), (3.14) e (3.15) do modelo matemático). A decodificação leva isso em consideração e ajusta a produção de P_1 para que seja concluída no micro-período anterior ao início da produção de P_2 em L_2.

Ao final do processo de decodificação, uma programação para linhas e tanques é estabelecida e sua avaliação é feita através da função de *fitness*. O valor do *fitness* é determinado pela função objetivo do modelo matemático:

$$\sum_{i=1}^{J}\sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{s=1}^{T\cdot S} s_{ijl}z_{ijls} + \sum_{j=1}^{J}\sum_{t=1}^{T} h_j I_{jt} + \sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{s=1}^{T\cdot S} v_{jl}q_{jls}$$
$$+ \sum_{i=1}^{\bar{J}}\sum_{j=1}^{\bar{J}}\sum_{k=1}^{\bar{L}}\sum_{s=1}^{T\cdot \bar{S}} \bar{s}_{ijk}\bar{z}_{ijks} + \sum_{j=1}^{\bar{J}}\sum_{k=1}^{\bar{L}}\sum_{t=1}^{T} \bar{h}_j \bar{I}_{jk,t\cdot T^m}$$
$$+ \sum_{j=1}^{\bar{J}}\sum_{k=1}^{\bar{L}}\sum_{s=1}^{T\cdot \bar{S}} \bar{v}_{jk}\bar{q}_{jks} + M\sum_{j=1}^{J} q_j^0$$

Nesse contexto, quanto menor o valor da função de *fitness*, mais adequado será o indivíduo já que a função objetivo busca minimizar custos.

6.2.4 Recombinação

A recombinação (*crossover*) implementada é do tipo ponto a ponto, ou seja, são avalidados genes na mesma posição em dois indivíduos diferentes. Dado o tipo de codificação aqui utilizada, a recombinação ponto a ponto foi implementada porque permite uma maior troca de informações durante o

processo de criação de um novo indivíduo.

Um exemplo é dado na figura 6.15 utilizando as demandas e produtos presentes na tabela 6.1.

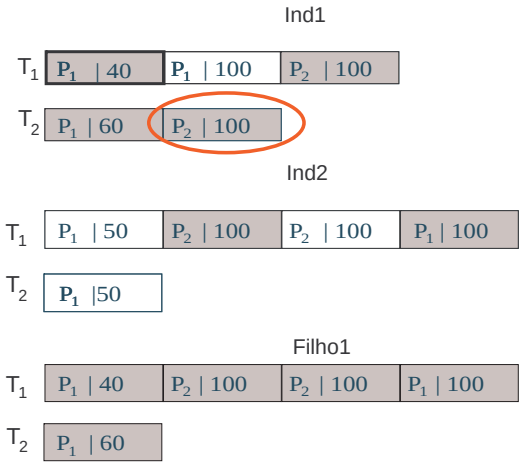

Fig. 6.15: Crossover - Filho1.

As informações relativas às seqüências de linhas e tanques (SL_{mn} e STk_{mn}) não aparecem na figura 6.15 porque não são relevantes. Essas seqüências são herdadas de um gene selecionado sem alteração pelo novo indivíduo (Filho1) e não tem influência na execução da recombinação. Os macro-períodos dos dois indivíduos (Ind1 e Ind2) são percorridos simultaneamente e os genes que ocorrem na mesma posição são avaliados. Para cada posição (gene) dentro de um macro-período em Ind1 e Ind2, um valor $\lambda \in [0, 1]$ é aleatoriamente gerado. Se $\lambda < 0.5$, Filho1 recebe o gene de Ind1; caso contrário, recebe o gene de Ind2. Os genes aleatoriamente selecionados estão sombreados na figura 6.15. Observe que o macro-período T_1 em Ind2 contém mais genes que em Ind1, levando o procedimento de recombinação a percorrer os genes de Ind2 até o final selecionando aqueles que satisfaçam $\lambda \geq 0.5$.

O processo de recombinação também verifica se as demandas estão sendo ultrapassadas durante a construção do novo indivíduo. Por exemplo, o gene dentro do círculo na figura 6.15 foi selecionado mas não foi inserido já que sua inserção ultrapassaria a demanda total do produto P_2. Se houvesse um gene na mesma posição em Ind2, este tomaria seu lugar em Filho 1 (desde que não incorresse no mesmo problema). Essa situação é melhor exemplificada na figura 6.16. O gene no círculo em Ind2 foi selecionado, porém sua inserção em Filho2 ultrapassaria a demanda total de P_1. Por isso, o gene na mesma posição em Ind1 ocupou seu lugar em Filho2.

A recombinação pode criar um indivíduo em que todas as demandas não tenham sido distribuídas pelo gene. Assim, um reparo é executado ao final da recombinação quando isso acontecer. A demanda faltante é inserida nos macro-períodos onde deixou de ser satisfeita. Essa situação está represenda na figura 6.17. A demanda total de P_1 em T_2 não foi completada em Filho3, levando a inserção de um

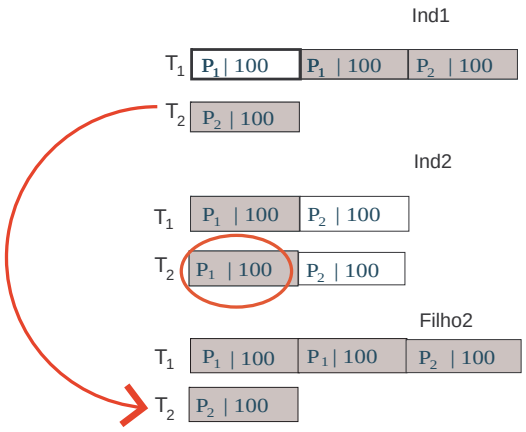

Fig. 6.16: Crossover - Filho2.

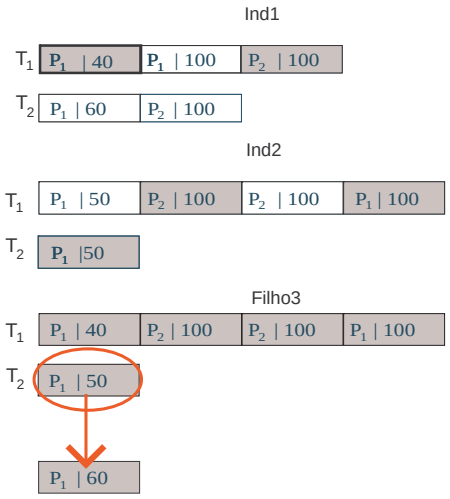

Fig. 6.17: Crossover - Filho3.

novo lote do produto.

6.2.5 Mutação

O operador genético de mutação procura manter certa diversidade na população, evitando uma convergência prematura para um tipo de indivíduo. Uma taxa de mutação é utilizada para determinar a quantidade de indivíduos alterados. Os indivíduos são selecionados aleatoriamente e as mutações executadas envolvem a troca de posição dos seus genes. Ao todo, três tipos de mutação foram utilizados e a figura 6.18 apresenta um exemplo para cada um deles.

O primeiro tipo de mutação altera a posição de dois genes dentro do mesmo macro-período. No segundo tipo, o gene selecionado é retirado da sua posição original e inserido em outra posição alea-

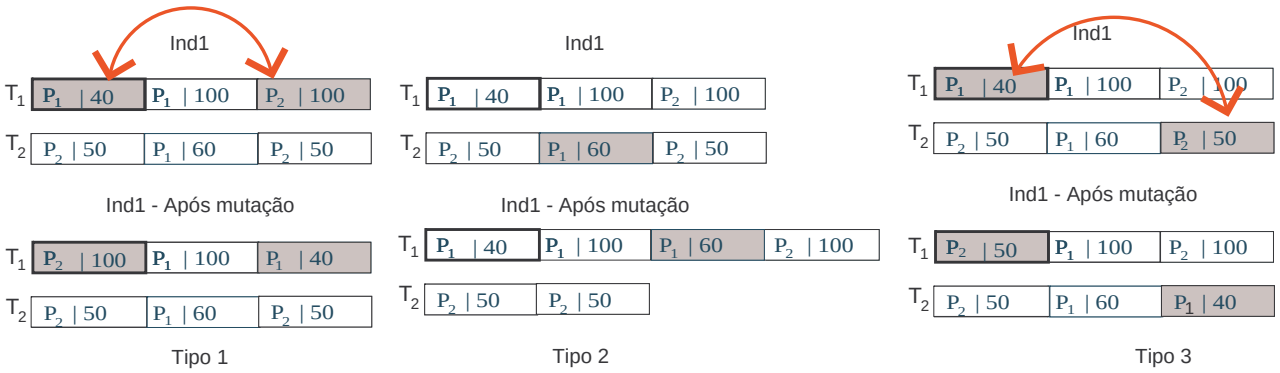

Fig. 6.18: Exemplo dos três tipos de mutação.

toriamente escolhida em um macro-período. O último tipo envolve a troca de posição entre dois genes em macro-períodos diferentes. Os genes e macro-períodos são aleatoriamente selecionados, mas alguns cuidados são levados em conta nas trocas envolvendo macro-períodos diferentes. A posição dos genes nos macro-períodos após as trocas deve continuar coerente com o macro-período das demandas dos produtos. Por exemplo, não se pode realizar mutações que posicionem um gene com demanda a ser atendida em T_1 no macro-período T_2. A demanda em T_1 deixa de ser satisfeita e tal mutação tornaria infactível a solução codificada no indivíduo (figura 6.19). Nesse caso, a mutação não acontece.

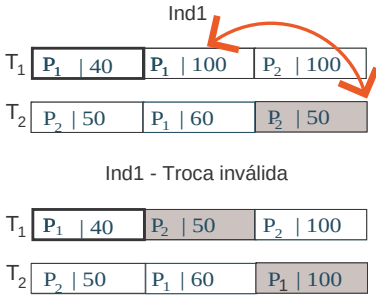

Fig. 6.19: Troca não permitida.

Observe na figura 6.19 que a demanda de P_1 deixou de ser atendida em T_1. Esse tipo de movimento não será executado. A cada execução do método **mutação**(novoInd) no algoritmo genético do Np-Opt, um dos três tipos de mutação é aleatoriamente selecionado para alterar o indivíduo.

6.2.6 Populações

O ambiente Np-Opt permite o uso de populações hierarquicamente estruturadas em árvores durante a execução dos algoritmos genético e memético. Os indivíduos são posicionados de acordo

com seu valor de *fitness* numa população estruturada hierarquicamente. O ambiente Np-Opt fornece dois tipos de estruturas: árvore binária e árvore ternária. A figura 6.20 apresenta um exemplo dessas estruturas. Também podemos utilizar uma população não estruturada no Np-Opt.

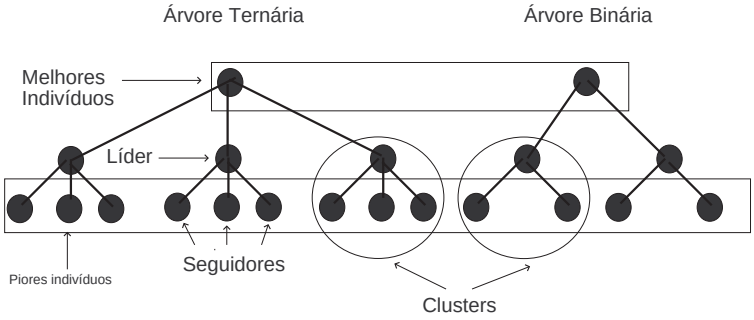

Fig. 6.20: Estrutura populacional.

As populações são separadas em *clusters* ou subpopulações. Na estrutura ternária, um *cluster* é composto por um líder (posicionado na raíz) e três seguidores (nós folhas). O *cluster* na árvore binária contém um líder e dois seguidores. França et al. (2001) resolvem o *total tardiness single machine scheduling problem* utilizando algoritmos genéticos e meméticos. Os resultados obtidos demonstraram um melhor desempenho tanto do AG quanto do AM com população estruturada. Mendes (2003) utiliza a estrutura ternária nos problemas avaliados e menciona que essa estrutura estabelece uma relação de hierarquia mais intuitiva, enquanto estruturas do tipo anel e malhas do tipo *grid* estabelecem apenas uma relação topológica entre indivíduos. Ele também argumenta, baseado em testes iniciais e considerações empíricas, que as árvores binárias apresentaram um "efeito subpopulacional"pequeno. Logo, a evolução simultânea de diferentes populações apresenta um impacto menor na busca por soluções em diferentes pontos do espaço de busca quando árvores binárias são usadas.

Considerando as observações anteriores, decidimos trabalhar com populações hierarquicamente estabelecidas dentro de uma árvore ternária. O processo de recombinação apresentado na seção anterior será agora detalhado no contexto desse tipo de estrutura. Inicialmente, o melhor indivíduo em um *cluster* é aleatoriamente selecionado e irá realizar a recombinação com um dos seus seguidores também aleatoriamente escolhido. O indivíduo filho será inserido no lugar de um dos pais, caso seu valor de *fitness* seja melhor. Um exemplo é dado na figura 6.21.

Um melhor indivíduo com valor de *fitness* 390 é aleatoriamente selecionado bem como seu seguidor com *fitness* de valor 570. O filho resultante dessa combinação apresenta um *fitness* de 100. O novo indivíduo será inserido no lugar do pai com pior valor de *fitness*. Lembramos que se trata de um problema de minimização, logo o pior *fitness* é aquele com maior valor numérico. A figura 6.22 apresenta o *cluster* após a inserção do novo indivíduo.

Uma fase de reestruturação da população ocorre após as recombinações. Todos os *clusters* são

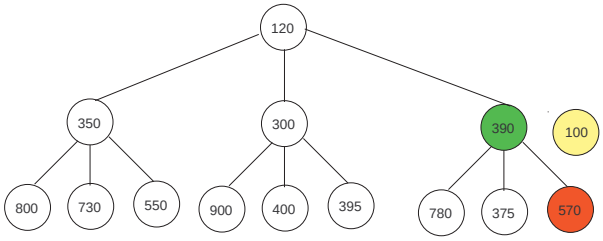

Fig. 6.21: População antes da inserção de novo indivíduo.

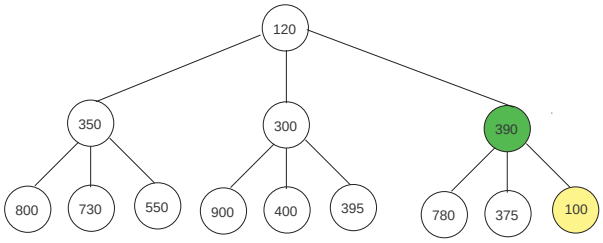

Fig. 6.22: População após inserção de novo indivíduo.

percorridos e se algum dos seguidores apresentar melhor *fitnes* que um líder, uma troca de posições irá ocorrer. A figura 6.23 ilustra esta idéia dentro do *cluster* mais interno na população e o processo segue a mesma idéia na população como um todo (figura 6.24).

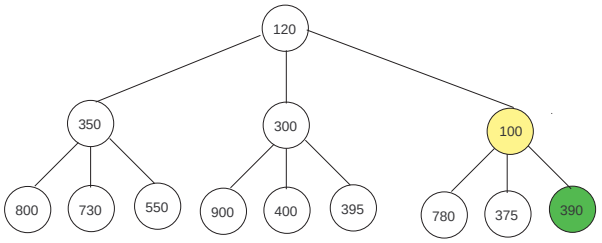

Fig. 6.23: Reestruturação das populações - Passo1.

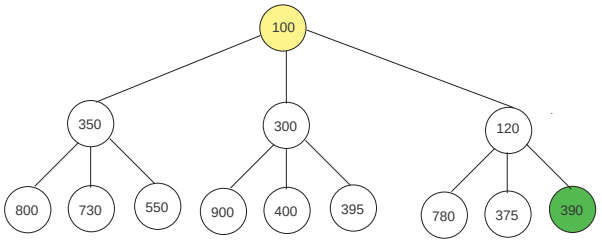

Fig. 6.24: Reestruturação das populações - Passo2.

Quando após uma série de recombinações não se consegue mais criar indivíduos com valor de *fitness* melhor que de seus pais, a população terá convergido e deve ser reinicializada. Antes de ser

reinicializada, o AG implementado no Np-Opt realiza uma migração entre as populações. Há três tipos de migração disponíveis:

- *0-Migrate*: nenhuma migração ocorre.
- *1-Migrate*: uma cópia do melhor indivíduo de cada população migra para a população seguinte.
- *2-Migrate*: duas cópias do melhor indivíduo de uma população migra para as duas populações seguintes.

Decidimos utilizar *1-Migrate* porque apresentou melhores resultados em testes realizados na fase de implementação do método. Além disso, Mendes (2003) indica que essa política migratória é a com melhor desempenho nos problemas por ele avaliado. A figura 6.25 ilustra a migração utilizada. Há

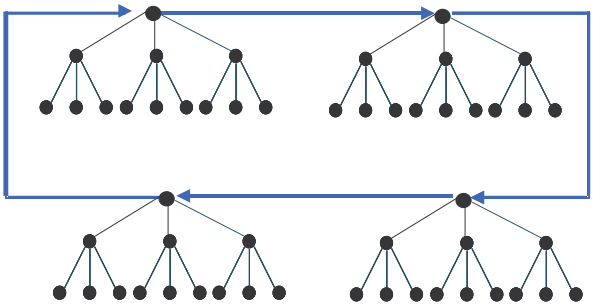

Fig. 6.25: *1-Migrate*.

quatro populações e cada uma é composta pelo melhor indivíduo e três *clusters*. As setas indicam a migração de uma cópia do melhor indivíduo para a população seguinte. Esse indivíduo é inserido no lugar de um indivíduo aleatoriamente selecionado, exceto o melhor da população. A reinicialização das populações poupa os melhores indivíduos e aqueles que acabaram de migrar.

6.3 Algoritmo Memético

6.3.1 Aspectos gerais

Algoritmos Meméticos (AM) são Algoritmos Genéticos (AG) que incorporam um procedimento de busca local(BL), permitindo intensificar a busca por soluções dentro da vizinhaça dos indivíduos no AG. A palavra memético origina-se de *meme* definido como a menor unidade de conhecimento capaz de ser transmitida (Moscato (1989)). O fluxograma da figura 6.26 apresenta em linhas gerais o AM implementado. A incorporação da busca local ao AG aqui apresentado irá atuar em indivíduos com um tipo complexo de codificação. Por isso, optamos por executá-la apenas no melhor indivíduo de cada população.

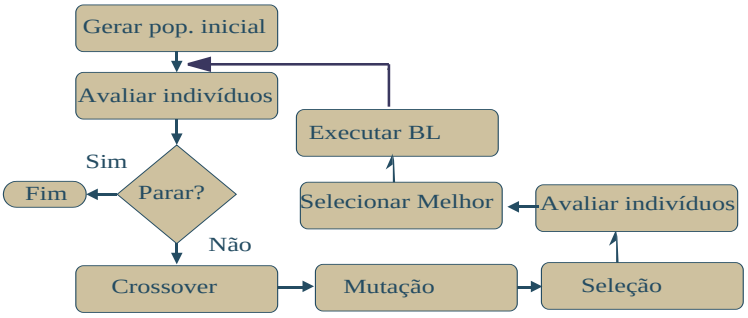

Fig. 6.26: Algoritmo Memético - Fluxograma

6.3.2 Busca Local

Um algoritmo chamado *threshold accepting* (TA) foi utilizado como base para o desenvolvimento de nossas buscas locais. O principal razão para escolha desse algoritmo foi sua utilização nos trabalhos de Meyr (2000) e Meyr (2002) que tratam, respectivamente, do Problema Geral de Dimensionamento de Lotes em uma única máquina e em máquinas paralelas. TA é utilizado nesses trabalhos para executar uma busca na vizinhança de atribuição dos produtos às linhas, ou seja, atua sobre a seqüência da programação estabelecida na linha. Além disso, TA é um algoritmo de busca local com desempenho superior ou igual à *Simulated Annealing* segundo Kirkpatrick et al. (1983) sendo mais fácil de implementar que uma Busca Tabu. Um algoritmo do tipo TA é apresentado abaixo:

begin

atualIndivíduo = Indivíduo;

repeat

novoIndivíduo = executaMovimento(atualIndivíduo);

Δf:=fitness(atualIndivíduo) - fitness(novoIndivíduo);

if ($\Delta f >$-Th.fitness(atualIndivíduo)) **then**

atualIndivíduo = novoIndivíduo;

if ($\Delta f \leq$-Th.fitness(atualIndivíduo)) **then**

reduzir(Th);

until númeroMáximoIterações;

end

O atualIndivíduo recebe inicialmente o melhor indivíduo de uma população e o novoIndivíduo recebe esse indivíduo após um movimento de busca em vizinhança. Se o *fitness* de novoIndivíduo

for melhor, novoIndivíduo passa a ser atualIndivíduo. Porém, TA evita convergência prematura ao permitir que soluções de pior qualidade sejam aceitas dentro do limiar estabelecido pelo parâmetro Th. Reduções constantes no valor de Th levam o método a convergir. Três aspectos da codificação do indivíduo são explorados na busca local proposta, ou seja, três tipos de movimentos para percorrer a vizinhança de um indivíduo serão executados:

1. Troca de genes
2. Alterações nos lotes
3. Alterações nas seqüências de linhas (α) e tanques (β).

A troca de genes visa provocar alterações na seqüência de programação dos produtos. Essa troca será executada através de dois possíveis movimentos:

ψ_1 : trocar a posição de dois genes dentro de um indivíduo.

ψ_2 : remover um gene de um macro-período e inserir em outro macro-período do indivíduo.

Os genes e macro-períodos envolvidos são escolhidos aleatoriamente. No caso das inserções, a posição onde um gene é inserido também é aleatoriamente selecionada. Temos aqui os mesmos movimentos utilizados para alterar indivíduos na mutação. Logo, os mesmos cuidados são tomados para impedir que a troca, remoção ou inserção de um gene leve ao não atendimento de demandas em um macro-período. As alterações nos lotes visam explorar outras possíveis distribuições das demandas nos indivíduos. Os valores para novos lotes são definidos pelos seguintes movimentos:

ϕ_1 : dividir o lote de um gene em dois lotes desiguais. Um dos lotes permanece no gene e o outro é incluído em uma nova posição dentro do indivíduo.

ϕ_2 : unir lotes de um mesmo produto em genes diferentes. Um gene é removido e seu lote é incorporado ao lote do mesmo produto em outro gene.

Também nos dois movimentos anteriores os genes, macro-períodos e posições (para inserção ou remoção) são aleatoriamente selecionados. As alterações nas seqüências de linhas (α) e tanques (β) são realizadas em um único movimento que trataremos por φ. Um gene selecionado terá seu primeiro par (α, β) alterado para outros valores aleatoriamente selecionados em $\alpha_i \in \{1, ..., L\}$ e $\beta_i \in \{1, ..., 2\overline{L}\}$. Decidimos alterar apenas o primeiro par (α, β) porque este será certamente utilizado no processo de decodificação do indivíduo. Todos esses movimentos foram separados gerando três tipos de busca local. Os movimentos ψ_1 e ψ_2 fazem parte do primeiro tipo de busca local que chamaremos de TA1. Da mesma forma, os movimentos ϕ_1 e ϕ_2 fazem parte de uma segunda busca local que

chamaremos de TA2. Tanto em TA1 como em TA2 a escolha do movimento a ser executado para alterar o indivíduo é feita de forma aleatória. A terceira busca local executa apenas o movimento φ e chamaremos de TA3. Todavia, TA3 segue o algoritmo TA com as seguintes alterações:

begin

atualIndivíduo = Indivíduo;

repeat

novoIndivíduo = executaMovimento(atualIndivíduo);

Δf:=fitness(atualIndivíduo) - fitness(novoIndivíduo);

if ($\Delta f >$-Th.fitness(atualIndivíduo)) **then**

atualIndivíduo = novoIndivíduo;

if ($\Delta f \leq$-Th.fitness(atualIndivíduo)) **then**

reduzir(Th);

until todos os genes de novoIndivíduos serem percorridos;

end

TA3 altera o primeiro par (α, β) de todos os genes do indivíduo partindo do primeiro macro-período até o último. Porém, a cada alteração o indivíduo é avaliado e será mantido se atender as condições do algoritmo TA.

O método de busca local existente no ambiente Np-Opt recebe o melhor indivíduo e executa TA1, TA2 e TA3 nessa ordem. Assim, as buscas vasculham os três tipos de vizinhanças e retornam o melhor indivíduo com as alterações que levaram a um melhor valor de *fitness*.

6.4 Resultados computacionais

6.4.1 Aspectos Gerais

Os resultados computacionais obtidos durante a solução das instâncias de pequena e elevada dimensão serão apresentados e analisados nesta seção. Todos os resultados aqui apresentados foram obtidos em um Pentium IV com 2.8 GHz.

A fórmula abaixo foi usada para medir o desvio (Desv) de uma solução em relação a outra que sirva como parâmetro de comparação:

$$Desv(\%) = 100\frac{(Z - \overline{Z})}{\overline{Z}} \tag{6.1}$$

onde, $\bar{Z}$ é o valor da solução usada como parâmetro e Z é o valor da solução avaliada.

O GAMS/Cplex foi executado apenas nas instâncias de pequena dimensão e durante o tempo limite de 1 hora, conforme mencionado no capítulo 5. O AG e AM tiveram seu tempo de execução ajustados de duas maneiras diferentes. No primeiro ajuste, uma execução de 360 segundos é repetida 5 vezes em cada instância onde o GAMS/Cplex levou até 1800 segundos para retornar a solução ótima. No segundo ajuste, uma execução de 1200 segundos é repetida 3 vezes em cada instância onde o GAMS/Cplex levou mais de 1800 segundos para retornar a solução ótima ou a melhor solução inteira factível. Esse último também foi o tipo de ajuste adotado para executar AG e AM nas instâncias de elevada dimensão. Repetir a execução mais de uma vez permitiu avaliar se a meta-heurística matém a qualidade da solução final.

O AG e AM também foram sempre executados com 3 populações hierarquicamente estruturadas em árvores ternárias. Cada população possui um total de 13 indivíduos. Assim, temos um nó lider correspondente ao melhor indivíduos e outros 3 *clusters* com seus respectivos líderes e seguidores. Alguns testes preliminares foram realizados com diferentes números de populações e indivíduos e os valores anteriores retornaram os melhores resultados.

O Algoritmo de busca local existente no AM é composto na realidade por três buscas locais (TA1, TA2 e TA3), conforme mencionado na seção 6.3.2. Cada uma dessas buscas foi ajustada para ser executada em um número máximo de 30 iterações com $Th = 0,20$ inicialmente. A cada melhoria no valor do *fitness* obtido pela busca local, o parâmetro Th é reduzido em 0,02. Caso o novo valor de *fitness* esteja apenas dentro do limiar de tolerância, Th é reduzido em 0,01. Se o novo valor do *fitness* estiver fora da tolerância, Th é reduzido em apenas 0,005.

6.4.2 Resultados para instâncias de pequena dimensão

As soluções obtidas pelo GAMS/Cplex nas instâncias de pequena dimensão serão usadas como parâmetro de comparação na avaliação do desempenho das meta-heurísticas propostas. Dois tipos de soluções foram obtidas pelo GAMS/Cplex: solução ótima ou a melhor solução inteira factível dentro do tempo limite de 1 hora de execução.

Nesta seção, poderemos availar se as meta-heurísticas conseguem determinar soluções ótimas ou soluções factíveis de boa qualidade. O objetivo aqui é avaliar se as meta-heurísticas conseguem alcançar a solução ótima nos conjuntos de instâncias onde o método exato foi capaz de determiná-las. Da mesma maneira, procuramos avaliar a qualidade da solução retornada pelas meta-heurísticas em relação às melhores soluções retornadas pelo método exato nos conjuntos de instâncias onde a solução ótima não foi alcançada.

Soluções ótimas foram retornadas pelo GAMS/Cplex em todos os conjuntos de instâncias onde $T = 1$. A tabela 6.2 apresenta o valor médio do Desv das soluções obtidas pelo AG e AM em relação

às soluções ótimas. A primeira coluna contém os parâmetros adotados em cada conjunto $L/\overline{L}/J/\overline{J}/$

	AG - Desv(%)			AM - Desv(%)		
$L/\overline{L}/J/\overline{J}$	Méd.	Máx.	Mín.	Méd	Máx	Mín
2 / 2 / 2 / 1	0,00	0,00	0,00	0,00	0,00	0,00
2 / 2 / 3 / 2	0,00	0,00	0,00	0,01	0,00	0,05
2 / 2 / 4 / 2	0,34	0,34	0,34	0,00	0,00	0,00
3 / 2 / 2 / 1	0,00	0,00	0,00	0,00	0,00	0,00
3 / 2 / 3 / 2	0,00	0,00	0,00	0,00	0,00	0,00
3 / 2 / 4 / 2	0,00	0,00	0,00	0,04	0,00	0,08
4 / 2 / 2 / 1	0,00	0,00	0,00	0,00	0,00	0,00
4 / 2 / 3 / 2	0,00	0,00	0,00	0,06	0,00	0,31
4 / 2 / 4 / 2	0,00	0,00	0,00	0,00	0,00	0,00

Tab. 6.2: Valores dos diversos Desv em relação à solução ótima em $T = 1$

com 10 instâncias criadas pelo gerador apresentado no capítulo 4. A coluna Méd apresenta o valor médio do desvio (Desv) considerando as soluções finais obtidas após todas as execuções do AG e AM. A coluna Máx apresenta o valor médio do desvio (Desv) considerando as piores soluções finais obtidas pelo AG e AM em cada execução. A coluna Mín apresenta o valor médio do desvio (Desv) considerando apenas as melhores soluções finais obtidas pelo AG e AM em cada execução.

As meta-heurísticas não encontram dificuldade em determinar soluções ótimas na grande maioria dos conjuntos de instâncias. Mesmo nos conjuntos onde existiu algum desvio, ele foi inferior a 1%. As soluções ótimas não são a totalidade dos resultados obtidos quando $T = 2$. Por isso a tabela 6.3 apresenta inicialmente o valor médio do desvio (Desv) entre as soluções obtidas pelo GAMS/Cplex, AG e AM em relação ao limitante inferior fornecido pelo algoritmo *branch & cut* do GAMS/Cplex.

	GAMS	AG - Desv(%)			AM - Desv(%)		
$L/\overline{L}/J/\overline{J}$	Desv(%)	Méd.	Mín.	Máx.	Méd	Mín	Máx
2 / 2 / 2 / 1	0,00	0,27	0,23	0,31	1,16	0,72	1,69
2 / 2 / 3 / 2	1,50	1,68	1,61	1,79	2,45	1,93	3,41
2 / 2 / 4 / 2	1,02	1,68	1,40	1,87	2,24	1,55	2,81
3 / 2 / 2 / 1	0,82	1,61	1,48	1,85	1,18	1,12	1,29
3 / 2 / 3 / 2	3,15	2,98	2,82	3,27	3,90	3,22	4,52
3 / 2 / 4 / 2	2,21	2,23	2,11	2,29	2,85	2,25	3,54
4 / 2 / 2 / 1	1,23	2,55	2,22	3,04	3,26	2,36	4,34
4 / 2 / 3 / 2	1,07	2,92	1,93	3,62	3,57	2,39	4,57
4 / 2 / 4 / 2	4,23	2,02	2,00	2,06	3,52	2,73	4,11

Tab. 6.3: Valores dos diversos Desv em relação ao limitante inferior em $T = 2$

A solução ótima é encontrada em todas as instâncias do conjunto 2/2/2/1 na coluna GAMS, já

que o valor médio de Desv é nulo. Os valores de Desv no AG ficam abaixo de 1% e AM apresenta comparativamente o pior desempenho em 2/2/2/1. Há um crescimento no valor de Desv quando o número de linhas e produtos aumentam. O AG encontra soluções com menos de 2% de desvio nos conjuntos onde $L = 2$ e menos de 3% na maioria das demais instâncias. O AM não consegue manter esses valores e apresenta uma piora em relação ao AG.

A figura 6.27 compara os valores de Desv presentes na coluna Méd em cada conjunto de instância. AG tem Desv inferior ao GAMS/Cplex nos conjuntos 3/2/3/2 e 4/2/4/2, o mesmo acontece com o

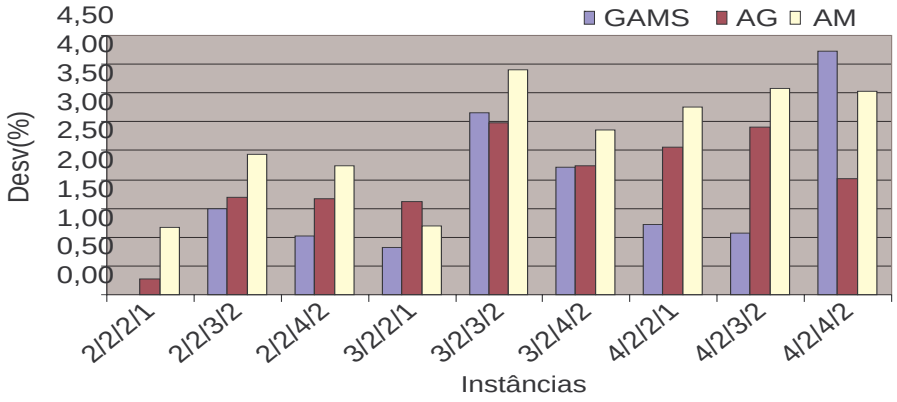

Fig. 6.27: Comparação do Desv Méd em relação ao limitante inferior em $T = 2$.

AM em 4/2/4/2. A tabela 6.4 compara o Desv entre as soluções de AG e AM em relação à solução final retornada pelo GAMS/Cplex.

$L/\overline{L}/J/\overline{J}$	AG - Desv(%)			AM - Desv(%)		
	Méd.	Mín.	Máx.	Méd	Mín	Máx
2 / 2 / 2 / 1	0,27	0,23	0,31	1,16	0,72	1,69
2 / 2 / 3 / 2	0,18	0,11	0,28	0,92	0,42	1,88
2 / 2 / 4 / 2	0,65	0,38	0,83	1,21	0,53	1,77
3 / 2 / 2 / 1	0,79	0,66	1,03	0,37	0,31	0,47
3 / 2 / 3 / 2	-0,11	-0,27	0,16	0,75	0,09	1,36
3 / 2 / 4 / 2	0,04	-0,08	0,10	0,60	0,01	1,27
4 / 2 / 2 / 1	1,30	1,01	1,72	1,81	1,05	2,72
4 / 2 / 3 / 2	1,84	0,86	2,53	2,48	1,31	3,47
4 / 2 / 4 / 2	-1,86	-1,88	-1,82	-0,39	-1,17	0,19

Tab. 6.4: Valores dos diversos Desv em relação à solução GAMS/Cplex em $T = 2$

Os valores negativos indicam que as soluções das meta-heurísticas superaram a qualidade das melhores soluções inteiras factíveis retornadas pelo GAMS/Cplex. Uma comparação entre os desempenhos das meta-heurísticas é apresentada na figura 6.28, onde observamos um melhor desempenho de AG na maioria das instâncias.

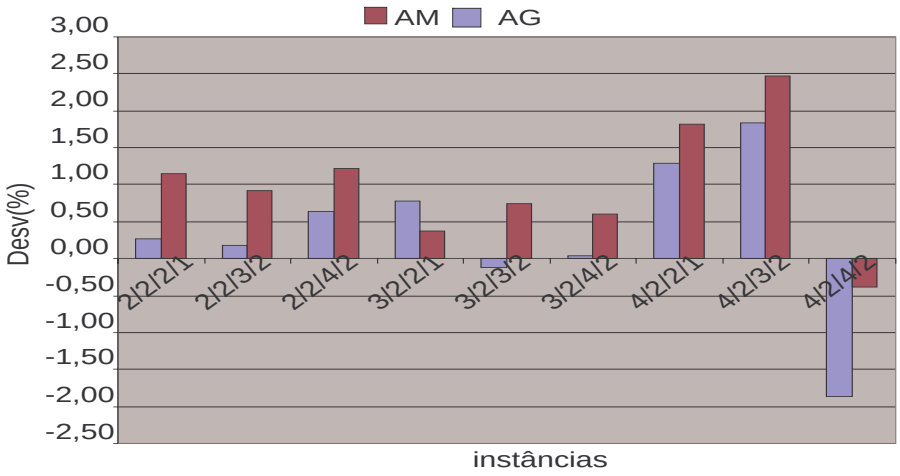

Fig. 6.28: Comparação do Desv Méd em relação à solução GAMS/Cplex em $T = 2$.

$L/\overline{L}/J/\overline{J}$	GAMS	AG - Desv(%)			AM - Desv(%)		
	Desv(%)	Méd.	Mín.	Máx.	Méd	Mín.	Máx
2 / 2 / 2 / 1	1,26	1,65	1,62	1,69	1,35	1,32	1,38
2 / 2 / 3 / 2	2,48	2,06	2,03	2,07	2,21	2,14	2,30
2 / 2 / 4 / 2	7,02	6,84	6,72	7,08	7,59	6,66	8,28
3 / 2 / 2 / 1	1,52	3,99	3,23	4,69	4,55	3,21	6,15
3 / 2 / 3 / 2	6,35	5,16	4,84	5,65	6,10	5,75	6,48
3 / 2 / 4 / 2	5,06	5,04	4,20	5,72	5,56	4,84	6,24
4 / 2 / 2 / 1	1,26	6,36	5,23	7,49	6,48	5,01	8,05
4 / 2 / 3 / 2	10,72	6,45	5,87	7,12	7,65	6,21	9,54
4 / 2 / 4 / 2	8,04	7,30	7,03	7,55	7,62	7,02	8,33

Tab. 6.5: Valores dos diversos Desv em relação ao limitante inferior em $T = 3$

A tabela 6.5 contém os valores do Desv em relação ao limitante inferior quando $T = 3$ e a figura 6.29 compara o desempenho dos métodos. Os valores dos Desv's aparecem próximos na maioria das instâncias, mas as meta-heurísticas começam a obter Desv Med abaixo do GAMS/Cplex em várias instâncias.

O desempenho do AG e AM em relação às soluções do método exato está na tabela 6.6 e na figura 6.30. AG e AM conseguem melhores soluções que o GAMS/Cplex em 5 conjuntos de instâncias, comprovando a dificuldade do método exato em retornar boas soluções quando L, J e T crescem. O AG continua obtendo soluções melhores que o AM.

A tabela 6.7 apresenta os resultados para $T = 4$ e a figura 6.31 compara o desempenho dos métodos. Os métodos apresentaram Desv abaixo de 10% na maioria dos conjuntos de instâncias. As meta-heurísticas passam a obter boa parte dos resultados mais próximos ao limitante inferior.

A tabela 6.8 e a figura 6.32 mostram o desvio em relação à solução GAMS/Cplex. As meta-

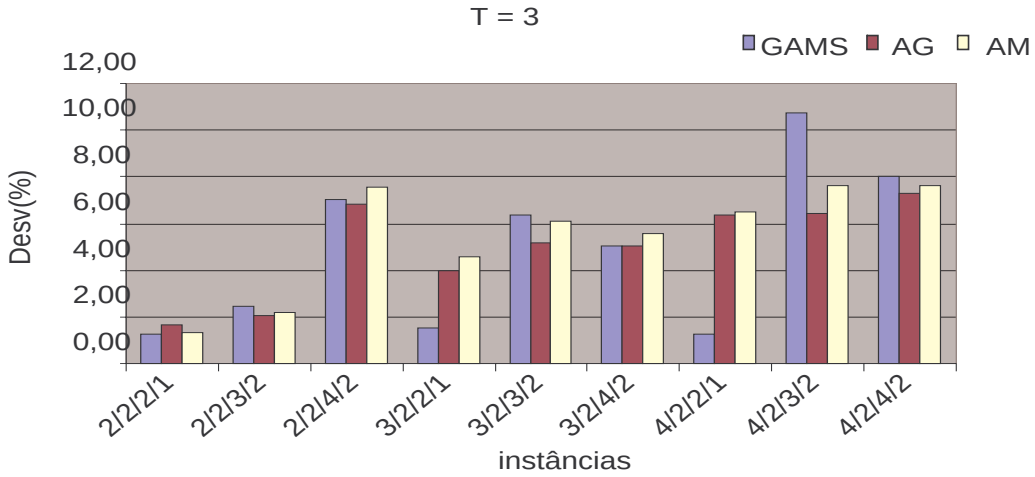

Fig. 6.29: Comparação do Desv Méd em relação ao limitante inferior em $T = 3$.

$L/\overline{L}/J/\overline{J}$	AG - Desv(%)			AM - Desv(%)		
	Méd.	Mín.	Máx.	Méd	Mín.	Máx
2 / 2 / 2 / 1	0,39	0,37	0,43	0,09	0,07	0,12
2 / 2 / 3 / 2	-0,40	-0,43	-0,39	-0,26	-0,32	-0,17
2 / 2 / 4 / 2	-0,73	-0,83	-0,53	-0,06	-0,88	0,56
3 / 2 / 2 / 1	2,26	1,52	2,95	2,81	1,50	4,39
3 / 2 / 3 / 2	-0,96	-1,27	-0,48	-0,04	-0,38	0,33
3 / 2 / 4 / 2	0,06	-0,74	0,70	0,56	-0,12	1,21
4 / 2 / 2 / 1	5,01	3,90	6,13	5,13	3,68	6,67
4 / 2 / 3 / 2	-3,62	-4,14	-3,01	-2,55	-3,83	-0,89
4 / 2 / 4 / 2	-0,56	-0,81	-0,34	-0,29	-0,83	0,36

Tab. 6.6: Valores dos diversos Desv em relação à solução GAMS/Cplex em $T = 3$

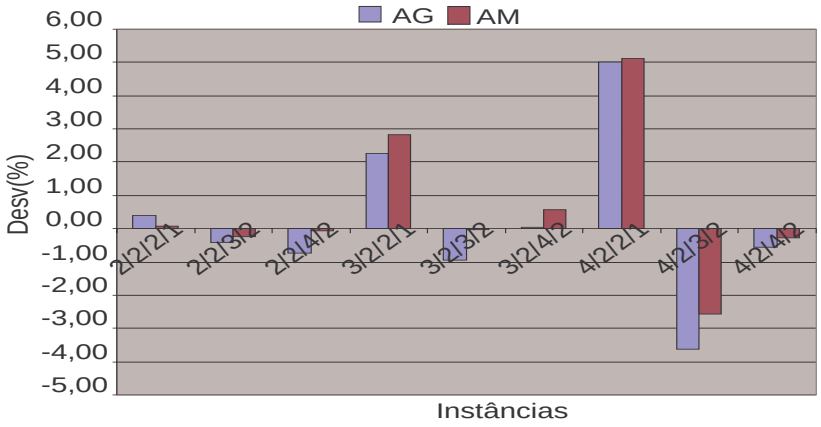

Fig. 6.30: Comparação do Desv Méd em relação à solução GAMS/Cplex em $T = 3$.

$L/\overline{L}/J/\overline{J}$	GAMS	AG - Desv(%)			AM - Desv(%)		
	Desv(%)	Méd.	Mín.	Máx.	Méd	Mín.	Máx
2 / 2 / 2 / 1	1,60	3,40	2,94	3,91	3,29	2,73	3,74
2 / 2 / 3 / 2	5,84	5,89	5,82	5,93	5,55	4,93	5,95
2 / 2 / 4 / 2	4,85	2,96	2,95	2,97	2,92	2,67	3,21
3 / 2 / 2 / 1	1,60	4,32	3,37	5,43	3,98	2,65	5,14
3 / 2 / 3 / 2	9,55	7,20	6,18	8,17	7,60	6,97	8,63
3 / 2 / 4 / 2	9,63	6,66	5,80	7,57	6,50	5,57	7,80
4 / 2 / 2 / 1	1,61	10,09	8,50	11,49	9,05	6,82	11,82
4 / 2 / 3 / 2	10,77	8,88	7,61	10,14	8,82	7,81	9,78
4 / 2 / 4 / 2	12,49	7,07	6,03	8,12	7,49	6,23	9,28

Tab. 6.7: Valores dos diversos Desv em relação ao limitante inferior em $T = 4$

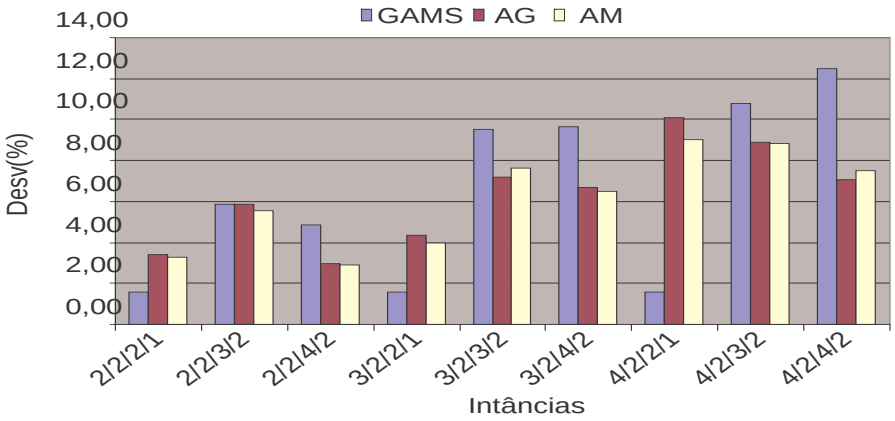

Fig. 6.31: Comparação do Desv Méd em relação ao limitante inferior em $T = 4$.

$L/\overline{L}/J/\overline{J}$	AG - Desv(%)			AM - Desv(%)		
	Méd.	Mín.	Máx.	Méd	Mín.	Máx
2 / 2 / 2 / 1	1,77	1,32	2,27	1,66	1,11	2,11
2 / 2 / 3 / 2	0,10	0,03	0,13	-0,26	-0,84	0,11
2 / 2 / 4 / 2	-1,63	-1,64	-1,62	-1,66	-1,89	-1,41
3 / 2 / 2 / 1	2,61	1,68	3,71	2,28	0,97	3,42
3 / 2 / 3 / 2	-2,41	-3,34	-1,51	-2,05	-2,63	-1,09
3 / 2 / 4 / 2	-2,56	-3,34	-1,75	-2,70	-3,54	-1,55
4 / 2 / 2 / 1	8,12	6,57	9,50	7,11	4,92	9,84
4 / 2 / 3 / 2	-2,22	-3,31	-1,13	-2,25	-3,15	-1,41
4 / 2 / 4 / 2	-3,59	-4,54	-2,63	-3,21	-4,37	-1,55

Tab. 6.8: Valores dos diversos Desv em relação à solução GAMS/Cplex em $T = 4$

heurísticas retornam soluções com melhor qualidade do que as soluções finais obtidas pelo GAMS/Cplex na grande maioria dos conjuntos de instâncias.

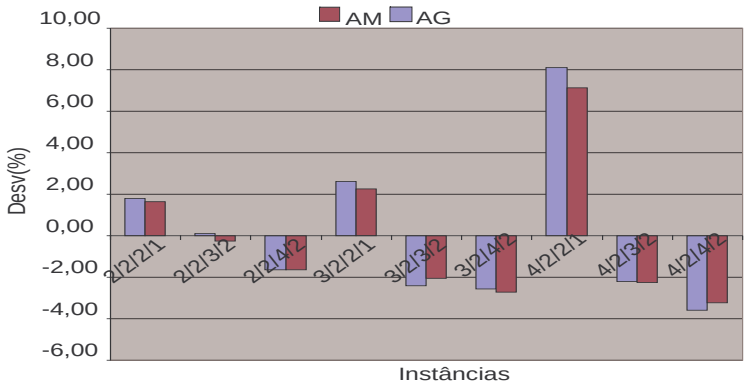

Fig. 6.32: Comparação do Desv Méd em relação à solução GAMS/Cplex em $T = 4$.

As figuras 6.33, 6.34 e 6.35 apresentam a evolução do valor médio do Desv Méd no decorrer dos macro-períodos. Os conjuntos são agrupados pelo mesmo valor de L. As meta-heurísticas retornam

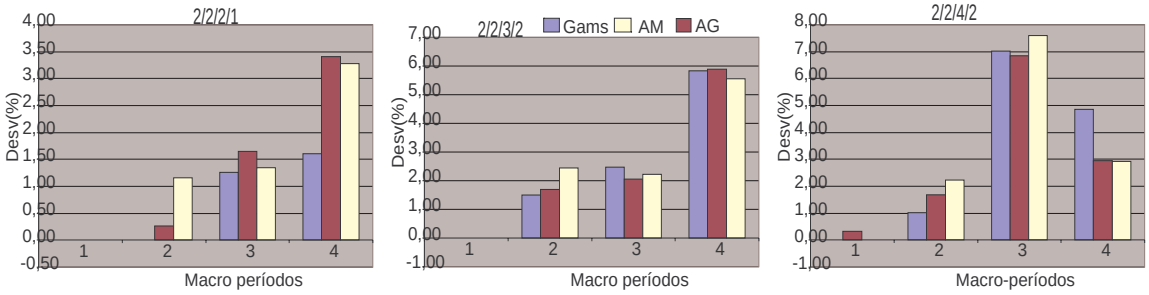

Fig. 6.33: Evolução do valor médio do Desv Méd. em relação ao limitante inferior quando $L = 2$

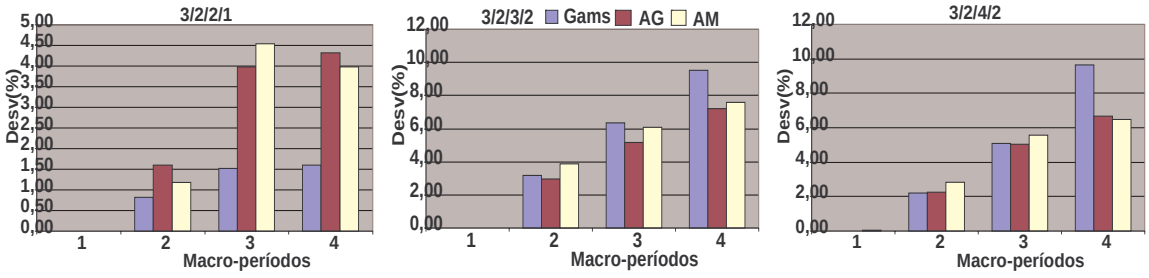

Fig. 6.34: Evolução do valor médio do Desv Méd. em relação ao limitante inferior quando $L = 3$

soluções com valor de Desv Méd semelhante ou melhor que o retornado pelo GAMS/Cplex nos dois últimos macro-períodos, exceto nos conjuntos mais simples: 2/2/2/1, 3/2/2/1 e 4/2/2/1. Ainda nos dois últimos macro-períodos, esse desempenho do AG e AM em relação ao método exato segue o crescimento nos valores do parâmetro J para um mesmo valor de L.

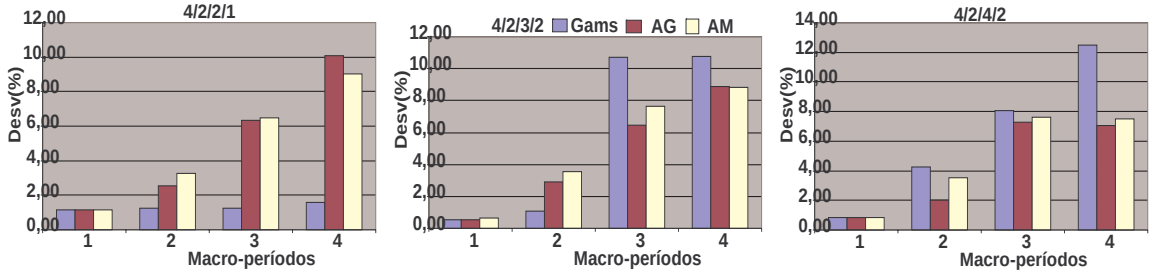

Fig. 6.35: Evolução do valor médio do Desv Méd. em relação ao limitante inferior quando $L = 4$

Há uma mudança considerável no patamar do Desv quando crescem os valores de L, J e T. Por exemplo, o valor do Desv Méd está abaixo de 5% em 3/2/2/1 e passa a oscilar abaixo de 10% em 3/2/4/2 na figura 6.34. Os métodos apresentaram Desv abaixo de $3,5\%$ no conjunto 2/2/2/1, o conjunto mais simples em termos de número de variáveis binárias do modelo matemático, e passam a oscilar abaixo de 14% no conjunto 4/2/4/2, o mais complexo segundo o mesmo critério.

Esse distanciamento do limitante inferior comprova que todos os métodos encontram dificuldades em retornar soluções finais de boa qualidade à medida que a complexidade do problema aumenta. O distanciamento ocorre de forma mais acelerada no GAMS/Cplex como mostrado nas figuras anteriores.

As figuras 6.36, 6.37 e 6.38 apresentam a evolução do Desv Méd em relação à solução GAMS/Cplex. Os gráficos demonstram que o crescimento no número de macro-períodos leva as meta-heurísticas a

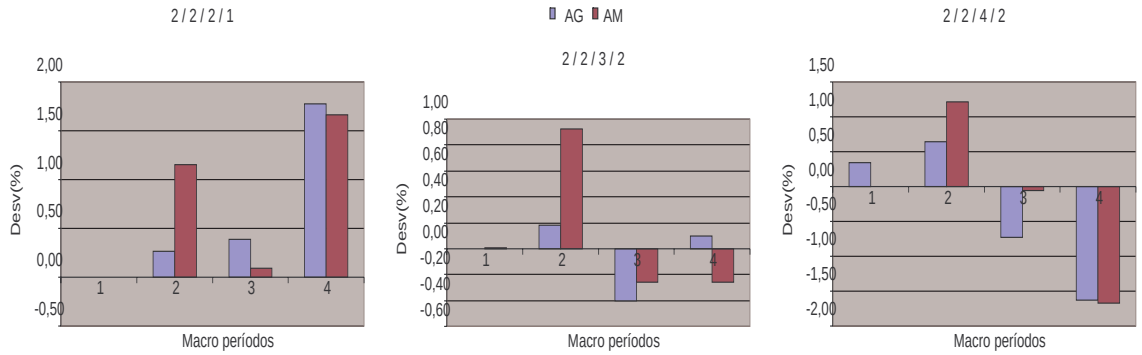

Fig. 6.36: Evolução do Desv Méd. em relação à solução GAMS/Cplex com $L = 2$

fornecerem soluções que superam àquelas retornadas pelo GAMS/Cplex. Isso fica evidente ao compararmos novamente o conjunto de instâncias mais simples 2/2/2/1 com o mais complexo 4/2/4/2. No primeiro conjunto, as meta-heurísticas apresentavam um Desv positivo abaixo de 2%. No último conjunto, as meta-heurísticas superam as soluções do GAMS/Cplex em praticamente todos os macro-períodos. O AG apresenta desempenho melhor que o AM na maioria dos conjuntos

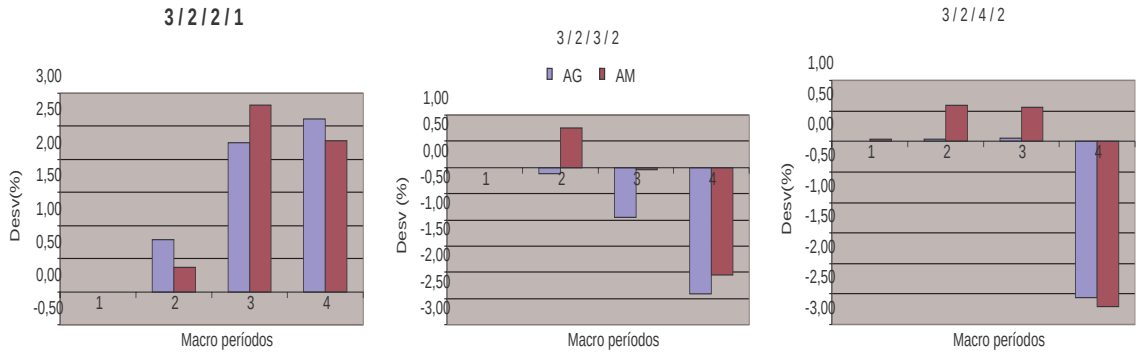

Fig. 6.37: Evolução do Desv Méd. em relação à solução GAMS/Cplex com $L = 4$

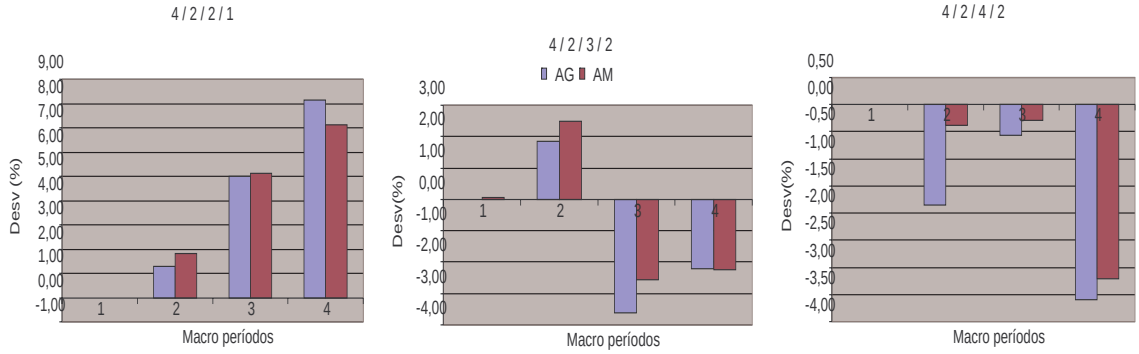

Fig. 6.38: Evolução do Desv Méd. em relação à solução GAMS/Cplex com $L = 4$

Podemos chegar a algumas conclusões nesta etapa. A maioria dos valores médios de Desv calculados (Méd, Mín e Máx) ficaram abaixo de 10% nas diversas comparações realizadas. O GAMS/Cplex apresentou a maior degradação no valor do Desv à medida que os parâmetros envolvidos cresciam. Isso se deve ao aumento no número de variáveis binárias nos repesctivos modelos matemáticos e à dificuldade encontrada por seu algoritmo *branch & cut* em determinar boas soluções dentro do tempo computacional fixado. Por outro lado, as meta-heurísticas conseguem retornar soluções mais próximas aos limitantes inferiores ao passo que os resultados do GAMS/Cplex pioram. Todos os métodos atestam a dificuldade em se resolver o PCDLPP mesmo considerando instâncias de pequena dimensão. Observa-se isso pelo reduzido número de soluções ótimas obtidas pelo método exato e pelo crescimento nos valores do Desv de todos os métodos. O AG apresentou desempenho melhor que o AM de um modo geral. A busca local pode estar guiando as melhores soluções avaliadas para ótimos locais, impedindo o método de explorar novos pontos no espaço de busca que o levem a melhores resultados.

6.4.3 Resultados para instâncias de elevada dimensão

O GAMS/Cplex não foi utilizado nas instâncias de elevada dimensão. A tentativa de utilizar o método em versões relaxadas do modelo matemático se mostrou inviável, conforme descrito no capítulo 5. Por isso, os resultados comparativos obtidos pelas meta-heurísticas é que serão analisados nessa seção. O objetivo aqui é avaliar a capacidade das meta-heurísticas para solucionar problemas com elevada dimensão dos parâmetros envolvidos. Assim, procura-se avaliar se os métodos conseguem determinar soluções dentro do tempo computacional fixado. Além disso, poderemos comparar qual método apresenta melhor desempenho.

A melhor solução encontrada nas 3 execuções realizadas em cada instância pelo AG foi utilizada como parâmetro de comparação. Todas as definições de Desv Méd, Máx e Mín continuam valendo aqui, mas tais Desv's foram calculados sempre em relação à melhor solução obtida pelo AG.

A tabela 6.9 apresenta os valores médios dos desvios quando $T = 4$. O AM consegue ganhos em termos de Desv em relação ao AG. Todas as soluções finais retornadas pelo AG não ficaram assim tão

$L/\overline{L}/J/\overline{J}$	AG - Desv(%)		AM - Desv(%)		
	Méd.	Máx.	Méd	Mín.	Máx
5 / 5 / 10 / 5	1,05	2,37	-0,10	-0,83	0,95
5 / 6 / 15 / 8	1,47	3,06	-6,21	-8,36	-3,40
8 / 5 / 10 / 5	3,69	9,78	0,72	-1,57	3,64
8 / 6 / 15 / 8	3,21	8,12	-0,79	-2,50	2,11

Tab. 6.9: Valores dos diversos Desv em relação à melhor solução retornada pelo AG em $T = 4$

distantes da sua melhor solução. Os valores do Desv Méd em AG são inferiores a $1,5\%$ nos conjuntos de instâncias com $L = 5$ e menores que $3,7\%$ nos conjuntos com $L = 8$. Considerando apenas a pior solução final retornada pelo AG nas 3 execuções realizadas, seu Desv Máx fica abaixo de $3,1\%$ quando $L = 5$ e menor que 10% quando $L = 8$. O Desv Méd do AM indica que todas as soluções finais do AM superaram o melhor valor encontrado pelo AG na maioria dos conjuntos de instâncias. A exceção é o conjunto 8/5/10/5, onde o Desv Méd do AM foi positivo. Considerando apenas as melhores soluções finais retornadas pelo AM, observamos que todas elas superaram a melhor solução do AG de acordo com os valores do Desv Mín no AM. A figura 6.39 compara o desvio de todas as soluções finais do AM e AG em cada conjunto de instância.

O AM apresenta um desempenho melhor que o AG em termos de Desv Méd na totalidade dos conjuntos de instâncias avaliados. Todavia, essa situação não se mantem quando $T = 8$.

Os valores da tabela 6.10 demonstram que o AM deixa de obter melhores resultados que o AG na grande maioria das instâncias. Ao dobrarmos o número de macro-períodos, as soluções finais obtidas pelo AM se afastam da melhor solução retornada pelo AG. A figura 6.40 permite avaliar melhor essa situação.

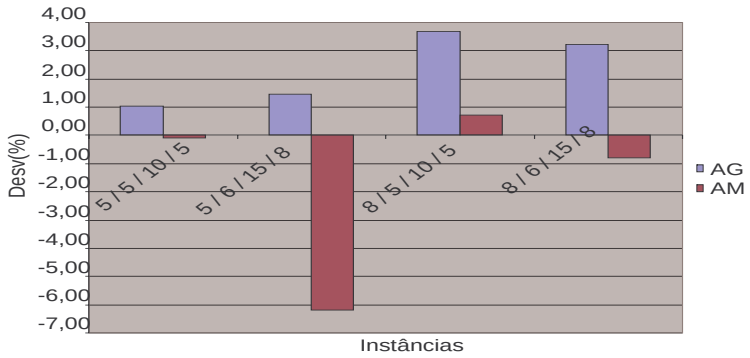

Fig. 6.39: Evolução do Desv Méd. em relação à melhor solução retornada pelo AG em $T = 4$

$L/\overline{L}/J/\overline{J}$	AG - Desv(%)		AM - Desv(%)		
	Méd.	Máx.	Méd	Mín.	Máx
5 / 5 / 10 / 5	1,86	4,27	2,15	-0,13	5,62
5 / 6 / 15 / 8	2,34	5,32	13,30	7,76	22,44
8 / 5 / 10 / 5	2,62	5,76	5,12	2,22	7,62
8 / 6 / 15 / 8	2,10	4,28	27,21	22,03	34,01

Tab. 6.10: Valores dos diversos Desv em relação à melhor solução retornada pelo AG em $T = 8$

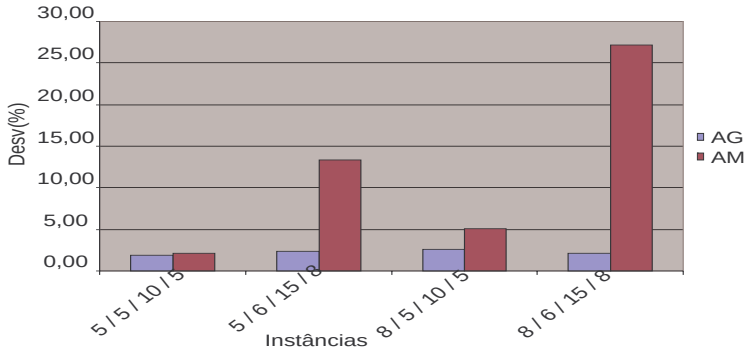

Fig. 6.40: Evolução do Desv Méd. em relação à melhor solução retornada pelo AG em $T = 8$

Ao passo que o AG apresenta um Desv Méd que não se altera consideravelmente de um conjunto para o outro, as soluções do AM se afastam cada vez mais da melhor solução do AG quando os valores de L e J aumentam. Os valores na tabela 6.11 apresentam os Desv's quando $T = 12$.

Os valores obtidos pelo AM aumentam sua distância da melhor solução encontrada pelo AG. Além disso, o desempenho do AM também se afastou mais do obtido pelo AG quando comparamos o Desv Méd de ambos os métodos. O crescimento no número de macro-períodos parece dificultar ainda mais a obtenção de boas soluções pelo AG associado à busca local aqui adotada (figura 6.41). O Desv Méd. no AG ficou abaixo de 5% e novamente não se alterou consideravelmente de um conjunto

$L/\overline{L}/J/\overline{J}$	AG - Desv(%)		AM - Desv(%)		
	Méd.	Máx.	Méd	Mín.	Máx
5 / 5 / 10 / 5	4,47	9,47	10,11	7,46	13,25
5 / 6 / 15 / 8	7,12	19,45	20,85	10,61	35,35
8 / 5 / 10 / 5	3,23	7,30	21,95	10,53	31,63
8 / 6 / 15 / 8	2,56	5,96	22,27	25,99	46,95

Tab. 6.11: Valores dos diversos Desv em relação à melhor solução retornada pelo AG em $T = 12$

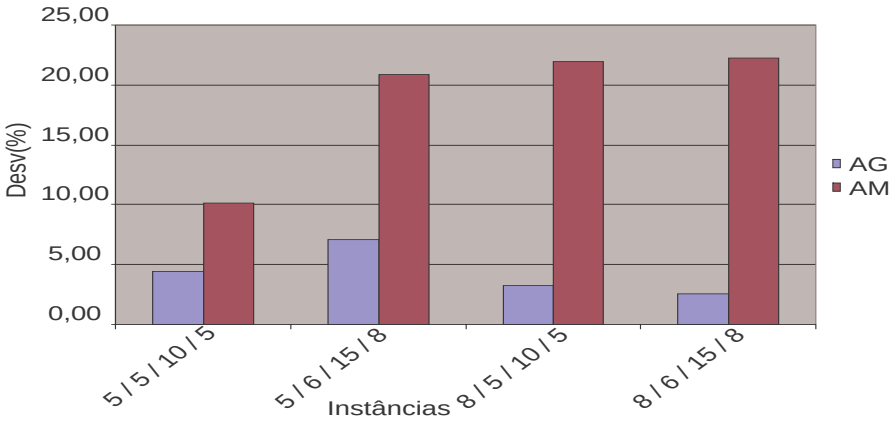

Fig. 6.41: Evolução do Desv Méd. em relação à melhor solução retornada pelo AG em $T = 12$

de instâncias para o outro. A exceção ocorreu em 5/6/15/8, onde o Desv. Méd ultrapassou 5%. O AM apresenta um crescimento no valor médio do Desv que acompanha o aumento do valor dos parâmentros nos conjuntos de instância.

A figura 6.42 apresenta a evolução do Desv Méd. do AG no decorrer dos macro-períodos. O

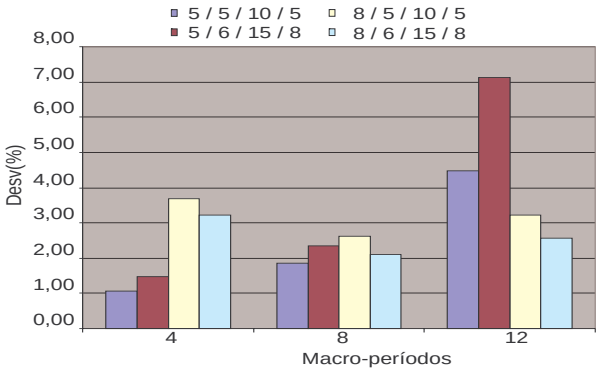

Fig. 6.42: Evolução do Desv Méd. do AG em relação à melhor solução retornada pelo AG em $T = 12$.

Desv Méd cresce no decorrer dos macro-períodos quando $L = 5$. Esse valor oscila quando $L = 8$, começando abaixo de 4% em $T = 4$, diminuindo em $T = 8$ e retornando a valores abaixo de 4% em

$T = 12$.

A figura 6.43 apresenta a evolução do Desv Méd do AM no decorrer dos macro-períodos. O

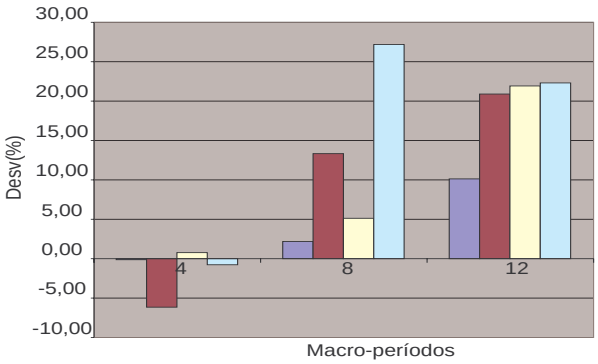

Fig. 6.43: Evolução do Desv Méd. do AM em relação à melhor solução retornada pelo AG em $T = 12$.

AM já apresenta um crescimento no valor do Desv Méd. no decorrer dos macro-períodos em todos os conjuntos de instâncias. Os valores de Desv Méd. no AG começam abaixo de 4% em $T = 4$ e terminam abaixo de 8% em $T = 12$. Porém, essa variação é bem maior no Desv Méd do AM que começa abaixo de 1% em $T = 4$ e acaba abaixo de 30% em $T = 12$.

Vamos também comparar o Desv Méd do AG e AM nas instâncias de maior complexidade quando $T = 4$ e $L = 2, 3, 4, 5$ e 8. A figura 6.44 demonstra que o crescimento no número de linhas leva a um

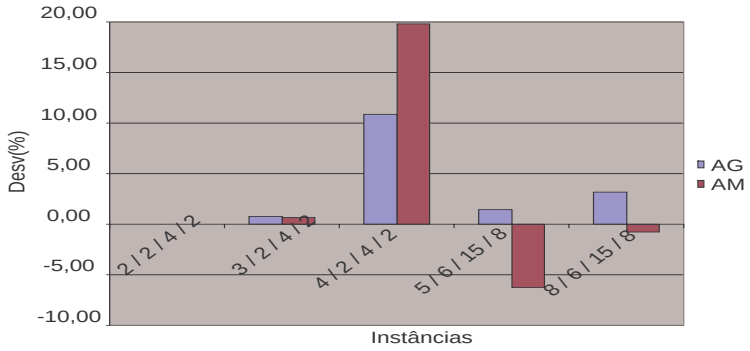

Fig. 6.44: Evolução do Desv Méd. nas instâncias mais complexas quando $T = 4$.

aumento no Desv Méd. A alteração no valor do Desv é maior no AM mesmo nas instâncias onde seu desempenho foi igual ou melhor que o AG. Assim, argumentamos que além do número de macro-períodos, o AM encontra maior dificuldade que o AG quando o número de linhas cresce. A associação do aumento no número de linhas com o aumento no número de macro-períodos compromete mais o desempenho do AM (6.45). O AM deixa de apresentar melhores valores de Desv Méd em $T = 4$, afastando-se melhores soluções obtidas pelo AG.

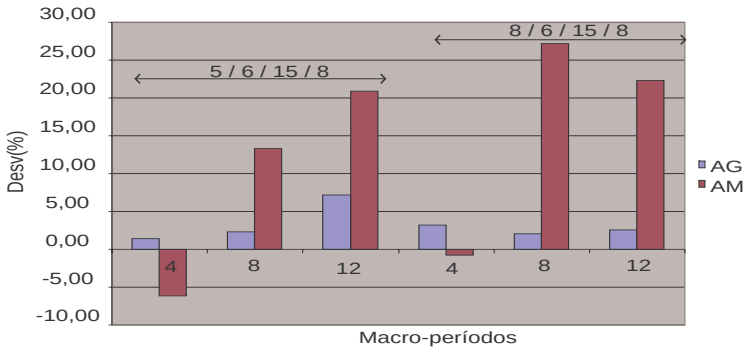

Fig. 6.45: Evolução do Desv Méd quando $J = 15$, $\overline{J} = 8$ e $\overline{L} = 6$.

A principal conclusão que podemos chegar nesta etapa é que o algoritmo de busca local adotado neste trabalho não melhora o desempenho do AG. Ao criarmos o AM adicionando ao AG uma busca local realizada sobre os melhores indivíduos, a meta-heurística acaba provavelmente se concentrando em vizinhanças que não levam ao encontro de melhores soluções. Por outro lado, o AG é uma alternativa bastante viável para solucionar problemas de elevada dimensão.

6.4.4 Resultados para instâncias industriais

As instâncias chamadas industriais foram criadas a partir de alguns dados fornecidos por uma indústria de bebidas. Essas informações fazem parte da programação realizada pela indústria durante 40 dias. A tabela 6.12 apresenta as instâncias e os valores usados nos respectivos parâmetros.

Instâncias	L	J	T	$\overline{L}$	$\overline{J}$	T^m
ind1	6	32	1	9	11	240
ind2	6	48	2	9	14	240
ind3	6	58	3	9	14	240
ind4	6	59	4	9	15	240

Tab. 6.12: Instâncias industriais

A programação se refere a um período em que não houve interrupção na produção, ou seja, a capacidade produtiva em 1 dia era de 24 horas. Cada macro-período T era composto por 10 dias e foi dividido em 240 micro-períodos T^m com capacidade $C^m = 1h$. As demandas devem ser atendidas ao final de cada período. Por exemplo, elas devem ser atendidas ao final do décimo dia em Ind1 e estão distribuídas no final do décimo e do vigésimo dia em Ind2. Os valores das demandas variam entre 200 e 200.000 unidades, dependendo do produto.

O cenário na indústria é composto por 6 linhas e 9 tanques. As linhas têm tempos de processamento que dependem do produto e variam entre 50 e 2200 unidades por hora. Há 6 tanques com

capacidade para 24000 litros de xarope, 2 tanques com capacidade para $22000l$ e apenas 1 tanque sendo continuamente reabastecido. Esse tanque foi considerado de capacidade infinita. Todos os demais parâmetros não mencionados variam de acordo com o intervalo de valores adotados no capítulo 4 para gerar instâncias de elevada dimensão.

As meta-heurísticas foram executadas usando o mesmo ajuste usado nas instâncias de elevada dimensão. Logo, 3 execuções de 1200 s. do AG e do AM foram realizadas em cada instância industrial. A melhor solução obtida ao final das 3 execuções pelo AG serve como parâmetro de comparação no cálculo dos diversos Desv's. A tabela 6.13 apresenta os valores dos Desv's para cada instância.

$L/\overline{L}/J/\overline{J}/T$	AG - Desv(%)		AM - Desv(%)		
	Méd.	Máx.	Méd	Mín.	Máx
6/9/32/11/1	0,21	0,33	-0,35	-0,40	-0,26
6/9/48/14/2	0,62	1,59	-1,68	-1,80	-1,47
6/9/58/14/3	0,96	1,70	-1,26	-1,88	-0,41
6/9/59/15/4	1,33	2,65	-1,48	-2,23	-0,52

Tab. 6.13: Valores dos diversos Desv de AG e AM nas instâncias industriais

O AM conseguiu melhorar a qualidade das soluções obtidas pelo AG em todas as instâncias industriais. Isso também pode ser verificado pela evolução dos valores do Desv Méd de AM e AG no decorrer dos macro-períodos (figura 6.46).

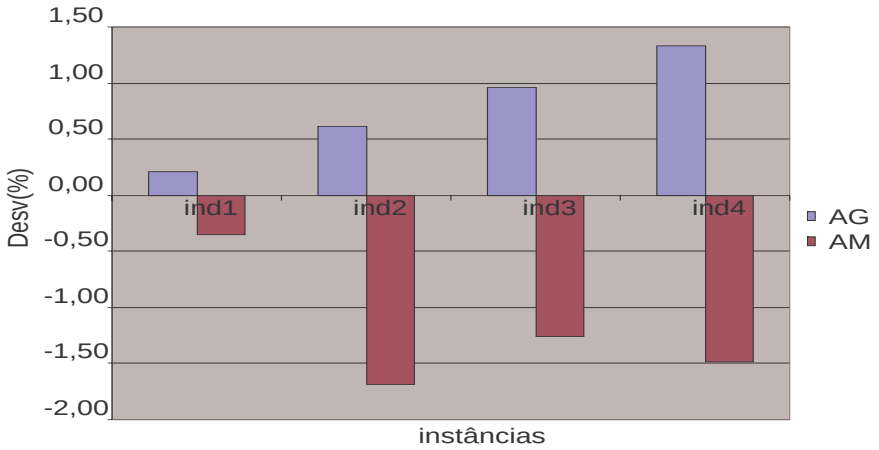

Fig. 6.46: Evolução do Desv Méd. em relação à melhor solução retornada pelo AG.

O Desv do AG cresce à medida que as instâncias aumentam em complexidade. O Desv Méd. oscila no AM, mas sempre com resultados que superam a melhor solução obtida pelo AG. Não podemos comparar as instâncias industriais com as instâncias de elevada dimensão. Alguns motivos que impedem tal comparação são as diferenças existentes em termos de capacidade dos tanques, número de micro-períodos, demanda dos produtos, entre outros.

Todavia, apenas para ilustrar, a figura 6.47 apresenta uma comparação envolvendo instâncias de elevada dimensão com $T = 4$ e a instância industrial $L/\overline{L}/J/\overline{J}/ = 6/9/59/15$ também com $T = 4$.

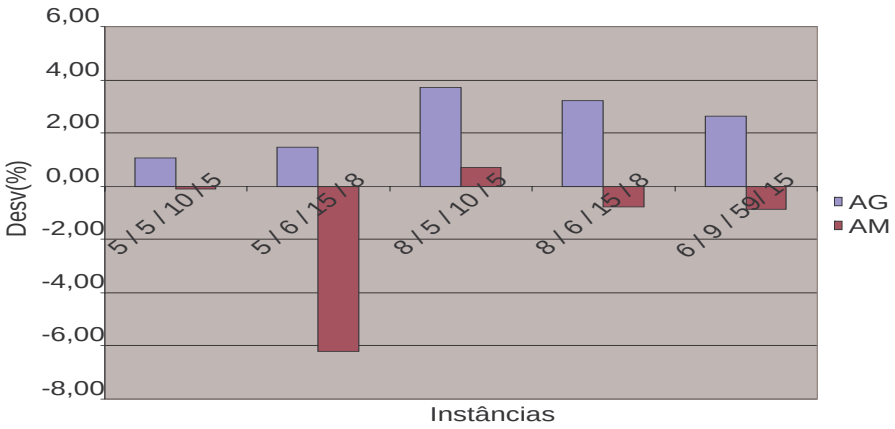

Fig. 6.47: Evolução do Desv Méd quando $J = 15$, $\overline{J} = 8$ e $\overline{L} = 6$.

O aumento no número de produtos parece não comprometer o desempenho do AM nessas instâncias. A meta-heurística apresentou um bom desempenho nos conjuntos de instâncias de elevada dimensão com $T = 4$ e esse desempenho não foi comprometido ao solucionar o problema industrial com o mesmo valor de T com um número bem maior de produtos.

6.5 Conclusão

O presente capítulo descreveu as meta-heurísticas propostas e apresentou os resultados computacionais obtidos. As meta-heurísticas foram implementadas dentro do ambiente de otimização Np-Opt e conseguiram utilizar diversas vantagems oferecidas por essa ferramenta: interface gráfica, reutilização de códigos, operadores genéticos, entre outros. As meta-heurísticas foram implementadas utilizando a estrutura multi-populacional hierarquica presente no Np-Opt. Todo esse tipo de estrutura pode ser diretamente incorporada aos nossos métodos devido às facilidades oferecidas pelo Np-Opt.

A codificação proposta procura inovar ao fugir das tradicionais representações binárias, inteiras ou regras simples de atribuições. Desenvolvemos uma codificação capaz de armazenar informações suficientes para permitir a criação de programações para linhas e tanques, ou seja, capaz de ser tranformada em uma solução do PCDLPP. O algoritmo de decodificação desempenha um papel importante neste processo. A principal vantagem na codificação adotada está na sua capacidade de facilitar a troca de informações através da recombinação genética. O gene contém uma gama de informações que são tranferidas imediatamente ao novo indivíduo durante a recombinação. Isso pode ser a resposta ao bom desempenho obtido pelo AG.

Por outro lado, a codificação adotada parece não facilitar a realização de uma busca local. O tipo de busca local aqui desenvolvido não conseguiu uma exploração no espaço de soluções tão boa quanto a recombinação genética, a partir da codificação adotada no indivíduo. O AG com busca local, ou seja, o AM não apresentou um desempenho tão satisfatória à medida que as instâncias cresciam em complexidade.

As duas meta-heurísticas apresentam um desempenho superior ao método exato nas instâncias de pequena dimensão. Elas encontram soluções ótimas ou se desviam pouco do valor ótimo onde eles ocorreram. Quando as soluções ótimas deixam de ser facilmente encontradas, as meta-heurísticas passam a ser mais eficientes na obtenção de soluções mais próximas ao limitante inferior. Os valores dos Desv's crescem em todos os métodos quando a complexidade das instâncias de pequena dimensão aumentam. Porém, tais valores permanecem abaixo de 10% do limitante inferior na maior parte dos conjuntos de instâncias. O AG continua superior ao AM nas instâncias de elevada dimensão conseguindo determinar melhores resultados na maior parte delas. Por outro lado, o AM tem desempenho superior ao AG nas instâncias industriais.

Capítulo 7

Conclusão

A presente tese propõe um novo problema chamado de Problema Conjunto de Dimensionamento de Lotes e Programação da Produção (PCDLPP). Trata-se de um problema multi-nível de dimensionamento de lote e programação da produção em máquinas paralelas com restrições de capacidade, custos e tempos de preparo dependentes da seqüência.

O problema combina aspectos raramente tratados em conjunto na literatura como tempos e custos de troca dependentes da seqüência e máquinas paralelas. Além disso, destacamos seu aspecto multi-nível ao estabelecer dois cenários interdependentes que exigem uma sincronia nas programações estabelecidas para cada um deles.

Um modelo matemático que capturasse todos os aspectos envolvidos no problema foi o primeiro desafio e a primeira contribuição deste trabalho. O modelo combinou aspectos presentes no Problema Geral de Dimensionamento e Programação de Lotes e no Problema Contínuo de Dimensionamento de Lotes.

Um número máximo de lotes foi determinado em cada macro-período. Esses lotes podem ser ocupados por diferentes produtos nas linhas e xaropes nos tanques. Variáveis binárias indexadas nos lotes passaram a representar a ocupação dos mesmos nos macro-períodos.

Os macro-períodos também foram divididos em micro-períodos de tamanho fixo. Os micro-períodos têm um papel fundamental na sincronia no tempo entre os dois níveis do problema. Através de variáveis contínuas e binárias indexadas pelos micro-períodos foi possível controlar, por exemplo, quando e quanto estava sendo produzido em cada estágio de produção do problema. Não encontramos abordagem similar na literatura que combinasse lotes e micro-períodos da forma aqui mencionada.

O modelo matemático apresentou uma complexa formulação que solicitou um vasto conjunto de parâmetros, variáveis e restrições. Acreditamos que tal modelo oferece um caminho novo e inicial em termos de modelagem desse tipo de problema. Todavia, apesar de sua representatividade do problema proposto, o modelo atual tem utilidade limitada para tratar instâncias de elevada dimensão. Isso

acontece devido à sua complexidade, portanto, simplificações do modelo podem ser uma pesquisa futura interessante.

A não existência de problemas semelhantes na literatura nos obrigou a desenvolver um método para gerar instâncias que nos permitisse avaliar o modelo proposto. Diversas fórmulas foram desenvolvidas e testadas. O objetivo era gerar instâncias factíveis e com um certo nível de utilização a partir de alguns parâmetros do problema.

A geração de instâncias foi uma tarefa árdua, já que o PCDLPP tem diversos fatores envolvidos. Por exemplo, determinada capacidade para o tanque limita a produção nas linhas. Por outro lado, determinado tempo de processamento na linha retarda as trocas nos tanques. O dilema constantemente enfrentado era a chance de se criar instâncias infactíveis ou muito simples de se resolver.

As fórmulas a que chegamos nesse trabalho permitiram principalmente reduzir o número de soluções infactíveis obtidas. Dois conjuntos de instâncias, que classificamos como de pequena e elevada dimensão, foram estabelecidos. Esses conjuntos a partir de agora podem servir para avaliação de outros métodos propostos para solucionar o PCDLPP.

Estabelecido o modelo matemático e criados conjuntos de instâncias, o passo seguinte foi encontrar soluções para essas instâncias. A primeira tentativa foi solucioná-las utilizando o pacote GAMS/Cplex dentro de um tempo limite de 1 hora. Os resultados computacionais nas instâncias de pequena dimensão demonstraram que pequenas mudanças no número de macro-períodos, produtos e linhas são suficientes para impedir a obtenção de soluções ótimas. O método exato encontrou na maioria das instâncias apenas a melhor solução inteira factível. Além disso, ele começa a retornar soluções com desvios cada vez maiores do limitante inferior quando a complexidade das instâncias aumenta.

As instâncias de elevada dimensão do PCDLPP simplesmente não podem ser solucionadas por um pacote como o GAMS/Cplex dentro do tempo limite estabelecido (1 hora). O desafio para o método exato foi encontrar ao menos uma solução factível para essas instâncias. A alternativa foi solucionar versões relaxadas do modelo, porém isso também se mostrou inviável. As variáveis menos indexadas permaneceram binárias e as demais passaram a ser contínuas, mas ainda assim o GAMS/Cplex encontrou dificuldades em retornar soluções factíveis. Isso frustrou nossa tentativa de estabelecer limitantes inferiores para o problema a partir de versões relaxadas do modelo matemático. Comparações desse tipo já havia sido realizada por Gupta and Magnusson (2005), mas não foi possível no PCDLPP.

O limitado alcance do método exato nos levou ao desenvolvimento de meta-heurísticas. O ambiente de otimização Np-Opt facilitou a implementação dessas meta-heurísticas. Um método baseado em Algoritmos Genéticos e outro em Algoritmos Meméticos foram desenvolvidos. Ambos utilizaram a estrutura hierarquica e multi-populacional presente no Np-Opt. Dessa forma, destacamos que o AG e AM aqui implemetados são multi-populacionais e estruturados em árvores ternárias hierarquica-

mente estabelecidas. Esse tipo de abordagem é inovadora em termos de meta-heurísticas.

Um tipo complexo de codificação foi desenvolvido para os indivíduos do AG e AM. O objetivo era criar uma codificação que armazenasse informações suficientes para o estabelecimento das programações nas linhas e tanques. Assim, o gene de um indivíduo passou a ter um peso importante no processo de decodificação e na troca de informações realizadas durante a recombinação genética.

As meta-heurísticas apresentaram um desempenho satisfatório nas instâncias de pequena dimensão. Elas foram capazes de encontrar as soluções ótimas, quando essas foram determinadas pelo GAMS/Cplex, na grande maioria das instâncias. Além disso, apresentaram desvios menores do que 10% do limitante inferior na maioria das instâncias. O AG e AM também retornaram soluções de melhor qualidade que o GAMS/Cplex à medida que as instâncias de pequena dimensão aumentavam em complexidade.

Nas instâncias de elevada dimensão, o AG obteve os melhores resultados. O método já apresentava um desempenho superior ao AM nas instâncias de pequena dimensão. Essa superioridade se destacou nas instâncias mais complexas. O AM apresentou dificuldades para retornar em média soluções melhores que o AG a medida que o número de macro-períodos e linhas aumentavam.

Algumas instâncias industriais foram criadas. Nessas instâncias, o AM confirmou o bom desempenho obtido em instâncias com menos macro-períodos ($T = 1, 2, 3$ e 4), superando o AG. Ambas as meta-heurísticas obtiveram bons resultados e foram capazes de encontrar soluções factíveis para esses problemas dentro de um tempo computacional relativamente curto.

Algumas propostas de pesquisas futuras são listadas a seguir:

- O modelo matemático proposto é um ponto de partida. Mudanças no modelo envolvendo redução de parâmetros, redução de variáveis e reformulação de restrições podem levar a formulações mais simples para o PCDLPP.

- Um gerador de instâncias que permita maior controle dos parâmetros envolvidos pode levar a diminuição do nível de instâncias infactíveis geradas.

- Outras codificações para os indivíduos podem ser desenvolvidas.

- Uma nova proposta de busca local que permita a melhoria do desempenho do AM. Novos movimentos podem ser tentados para melhorar a busca na vizinhança dos indivíduos.

- Aplicação das abordagens apresentadas em estudo de casos reais para verificar suas efetividades em relação às soluções utilizadas pelas empresas.

Referências Bibliográficas

R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows*. Prentice-Hall, 1993.

A.I. Ali, R. Padman, and H. Thiagarajan. Dual algorithms for pure network problems. *Operations Research*, 37(1):159–171, 1989.

V.A. Armentano, P.M. França, and F.M.B. Toledo. A network flow model for the capacitated lot-sizing problem. *OMEGA, The Internatinal Journal of Management Sciences*, 27:275–284, 1999.

H.C. Bahl, L.P. Ritzman, and J.N.D. Gupta. Determining lot sizes and resource requirements: a review. *Operations Research*, 35:329–345, 1987.

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

T. Baker and J.A. Muckstadt. The ches problems. Technical report, Chesapeake Decision Sciences, Inc., 1989.

I. Barany, T.J. van Roy, and L.A. Wolsey. Strong formulations for multi-item capacitated lotsizing. *Management Science*, 30:1255–1261, 1984.

T. Bäck, D.B. Fogel, and T. Michalewicz, editors. *Evolutionary Computation1, Basic Algorithms and Operators*. Insitute of Physics Publishing, 2000a.

T. Bäck, D.B. Fogel, and T. Michalewicz, editors. *Evolutionary Computation2, Advanced Algorithms and Operators*. Insitute of Physics Publishing, 2000b.

R.E. Berreta. *Heurísticas para Otimização do Planejamento da Produção em Sistemas MRP*. Tese de doutorado, Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação, Abril 1997.

G.R. Bitran and H. Matsuo. Approximation formulations for the single-product capacitated lot size problem. *Operations Research*, 34:63–74, 1986.

G.R. Bitran and H.H. Yanasse. Computational complexity of the capacited lot size problem. *Management Science*, 28:1174–1186, 1982.

J. Blazewicz, W. Domschke, and E. Pesch. The job shop scheduling problem: conventional and new solution techniques. *European Journal of Operational Research*, (93):1–33, 1996.

W. Bruggemann and H. Jahnke. The discrete lot-sizing and scheduling problem: complexity and modification for batch availability. *European Journal of Operational Research*, pages 511–528, 2000.

J.J. Carreno. Economic lot scheduling for multiple products on parallel identical processors. *Management Science*, 36:348–358, 1990.

W.H. Chen and J.M. Thizy. Analysis of relaxations for the multi-item capacitated lot-sizing problem. *Annals of Operations Research*, pages 29–72, 1990.

A.R. Clark and S.J. Clark. Rolling-horizon lot-sizing when set-up times are sequence-dependent. *International Journal of Production Research*, 38(10):2287–2307, 2000.

L. Davis, editor. *Handbook of genetic algorithms*. van Reinhold, 1991.

S.G. Davis. Scheduling econimc lot size production runs. *Management Science*, 36(8), 1990.

G. Dobson. The economic lot scheduling problem: achieving feasibility using time varying lot sizes. *Operations Research*, 35:764–771, 1987.

U. Dorndorf and E. Pesch. Evolution based learning in job shop scheduling enviroment. *Computer & Operations Research*, (22):25–40, 1995.

A. Drexl and K. Haase. Propotional lotsizing and scheduling. *International Journal of Production Economics*, 40:73–87, 1995.

A. Drexl and K. Haase. Sequential-analysis based randomized-regret-methods for lot-sizing and scheduling. *Journal of the Operational Research Society*, 47:251–265, 1996.

A. Drexl and A. Kimms. Lot sizing and scheduling - survey and extensions. *European Journal of Operational Research*, 99:221–235, 1997.

Y. Dumas, J. Desrosiers, E. Gelinas, and M.M. Solomon. An optimal algorithm for the travelling salesman problem with time windows. *Operations Research*, 43(2):367–371, 1995.

S.E. Elmaghraby. The economic lot scheduling problem (elsp): review and extensions. *Management Science*, 24(6), 1978.

D. Erlenkotter and F. W. Harris. The economic order quantity model. *Operations Research*, 38: 937–946, 1990.

B. Fleischmann. The discrete lot-sizing and scheduling problem. *European Journal of Operational Research*, 44:337–348, 1990.

B. Fleischmann and H. Meyr. The general lotsizing and scheduling problem. *OR Spektrum*, 19:11–21, 1997.

P.M. França, A.S. Mendes, and P.Moscato. A memetic algorithm for the total tardiness single machine scheduling problem. *European Journal of Operational Research*, 1(132):224–242, 2001.

G.D.Eppen and R.K. Martin. Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research*, 35:832–848, 1987.

F. Glover. Tabu-search-parti. *ORSA Jounal on Computing*, 1(3):190–206, 1989.

D.E. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison Wesley, 1989.

D. Gupta and T. Magnusson. The capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times. *Computer & Operations Research*, 32:727–747, 2005.

K. Haase. Capacitated lot-sizing with linked production quantities of adjacent periods. *Working Paper*, 334, 1993. University of Kiel.

K. Haase. Capacitated lot-sizing with sequence dependent setup costs. *OR Spektrum*, (18):51–59, 1996.

K. Haase and A. Kimms. Lot sizing and scheduling with sequence-dependent setup costs and times and efficient rescheduling opportunities. *International Journal of Production Economics*, (66): 159–169, 2000.

F.W. Harris. How many parts to make at once. *Operations Research*, 38:947–950, 1990.

K.S. Hindi. Solving the clsp by a tabu search heuristic. *Journal of the Operational Research Society*, 47:151–161, 1996.

K.S. Hindi, K. Fleszar, and C. Charalambous. An effective heuristic for the clsp with set-up times. *Journal of the Operational Research Society*, 54:490–498, 2003.

J.H. Holland. *Adaptation in natural and artificial systems*. The University of Michigan Press, 1975.

W.L. Hsu. On the general feasibility test of scheduling lot sizes for several products on one machine. *Management Science*, 29:93–105, 1983.

S. Kang, K. Malik, and L.J. Thomas. Lotsizing and scheduling on parallel machines withe sequence-dependent setup costs. *Management Sciences*, 45(2):273–289, 1999.

U.S. Karmakar and S. Kekre. The deterministic lotsizing problem with startup and reservation costs. *Operations Research*, 35:389–398, 1987.

A. Kimms. Multi-level lot sizing and scheduling-methods for capacitated, dynamic, and deterministic models. *Heidelberg:Physica*, 1997.

A. Kimms. A genetic algorithm for multi-level, multi-machine lot sizing and scheduling. *Computers & Operations Research*, 26:829–848, 1999.

Ö. Kirca and M. Kökten. A new heuristic approach for the multi-item dynamic lot sizing problem. *European Journal of Operational Research*, 75:332–341, 1994.

S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220 (4598):671–680, 1983.

R. Kuik, M. Salomon, and L.N. van Wassenhove. Batching decisions: structure and models. *European Journal of Operational Research*, 75:243–263, 1994.

R. Kuik, M. Salomon, L.N. Van Wassenhove, and J. Maes. Linear programming, simulated annealing and tabu search heuristics for lotsizing in bottleneck assembly systems. *IIE Transactions*, 25(1): 62–72, 1993.

J. Maes, J.O. McClain, and L.N. Van Wassenhove. Multilevel capacitated lotsizing complexity and lp-based heuristics. *European Journal of Operational Research*, 53:131–148, 1991.

J. Maes and L.N. van Wassenhove. Multi-item single level capacitated dynamic lot-sizing heuristics: a general review. *Journal of the Operational Research Society*, 39:991–1004, 1988.

A.S. Mendes. *O Framework NP-Opt e suas Aplicações a Problemas de Otimização*. PhD thesis, Universidade Estadual de Campinas, 2003.

H. Meyr. Simultaneous lotsizing and scheduling by combining local search with dual reoptmization. *European Journal of Operational Research*, 120:311–326, 2000.

H. Meyr. Simultaneous lotsizing and scheduling for parallel production lines. *European Journal of Operational Research*, 139:277–292, 2002.

Z. Michalewicz. *Genetic Algorithms + data structure = evolution programs*. Springer-Verlag, 1996.

P. Moscato. On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms. Technical Report C3P 826, Caltech Concurrent Computational Program, 1989.

M. Pinedo. *Scheduling - Theory, Algorithms, and Systems*. Prentice-Hall,Englewood Cliffs,NJ, 1995.

M. Salomon, L.G. Kroon, R. Kuik, and L.N. Van Wassenhove. Some extensions of the discrete lotsizing and scheduling problem. *Management Science*, 37:801–812, 1991.

M. Salomon, M. Solomon, L. N. Van Wassenhove, Y. Dumas, and S. Dauzére-Péres. Solving the discrete lotsizing and scheduling problem with sequence dependent set-up costs and set-up times using the travelling salesman problem with time windows. *European Journal of Operational Research*, 100:494–513, 1997.

V.L. Smith-Daniels and D.E. Smith-Daniels. A mixed integer programming model for lot sizing and sequencing packaging lines in the process industries. *IIE Transaction*, 18:278–285, 1986.

J.M. Thizy and L.N. van Wassenhove. Lagrangean relaxation for the multi-item capacity lot-sizing problem: A heuristic implementation. *IIE Transaction*, 17:308–313, 1985.

F.M.B. Toledo. *Dimensionamento de Lotes em Máquinas Paralelas*. Tese de doutorado, Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação, 1998.

W.W. Trigeiro, L.J. Thomas, and J.O. McClain. Capacitated lot sizing with setup times. *Management Science*, 35:353–366, 1989.

A. Wagelmans, S. van Hoesel, and A. Kolen. Economic lot sizing: an o(nlogn) algorithm that runs in linear time in the wagner-whitin case. *Operations Research*, (40):S145–S156, 1992.

H.M. Wagner and T.M. Whitin. Dynamic version of the economic lot size model. *Management Science*, 5:89–96, 1958.

J. Weiner. *The beak of the finch*. Vintage Books, New York, 1995.

L. Özdamar and S. I. Birbil. Hybrid heuristic for the capacitated lot sizing and loading problem with setup times and overtime decisions. *European Journal of Operational Research*, pages 525–547, 1998.

Linet Özdamar and Gülay Barbarosoglu. An integrated lagrangean relaxation-simulated annealing approach to the multi-level multi-item capacited lot sizing problem. *International Journal of Production Economics*, 68:319–331, 2000.

Linet Özdamar and Mehmet Ali Bozyel. The capacitated lot sizing problem with overtime decisions and setup times. *IIE Transactions*, 32:1043–1057, 2000.

P. Zipkin. Computing optimal lot sizes in the economic lot scheduling problem. *Operations Research*, 39:56–63, 1991.

Apêndice A

Heuristica de Fixação de Variáveis (HFV)

A.1 Introdução

Uma heurística baseada em fixação e relaxação de variáveis binárias também foi implemetada durante o desenvolvimento desta tese. Infelizmente, os resultados obtidos não foram satisfatórios e por essa razão foram incluídos neste apêndice. O método segue em linhas gerais o seguinte algoritmo:

Heuristica de Fixação de Variáveis - HFV

$cpu = 0$;$t = T$

while ($cpu < 3600$) e ($t > 0$)**do**

For $\overline{ss} = \overline{S}, \overline{S} - 1, ..., 1$ **do**;

For $ss = S, S - 1, ..., 1$ **do**;

Resolva o problema de Programação Inteira-Mista Relaxado(PIMR) com x_{jls}, u_{ls}, $x^B_{ls\tau}$, $x^E_{ls\tau}$, $\bar{x}_{jks1}$, $\bar{u}_{ks}$, $\bar{x}^E_{ks\tau}$, $\bar{x}^B_{ks\tau}$ binárias quando $s = ss$ e $\overline{s} = \overline{ss}$ e contínuas quando $s \neq ss$ e $\overline{s} \neq \overline{ss}$.

Fixar as variáveis binárias x_{jls}, u_{ls}, $x^B_{ls\tau}$, $x^E_{ls\tau}$ da solução obtida nos valores 0 e 1 encontrados.

$cpu = cpu+$tempo gasto na resolução do PIMR.

end

Fixar as variáveis binárias $\bar{x}_{jks1}$, $\bar{u}_{ks}$, $\bar{x}^E_{ks\tau}$, $\bar{x}^B_{ks\tau}$ da última solução obtida nos valores 0 e 1 encontrados.

end

$t = t - 1$.

end

O modelo matemático é relaxado de modo que poucas variáveis permaneçam binárias. O algoritmo percorre cada macro-período t do fim para o início. Para cada macro-período t, os lotes dos tanques existentes em t são percorridos a partir do último até o primeiro lote. Da mesma forma, para cada lote $\overline{ss}$ do tanque, todos os lotes das linhas são percorridos também do último para o primeiro. O modelo é solucionado mantendo como binárias apenas as variáveis indexadas nos lotes $\overline{ss}$ e ss.

As variáveis binárias das linhas são fixadas nos valores 0 e 1 da solução encontrada pelo PIMR para cada lote ss avaliado. Ao percorrer todos os lotes das linhas, a variável binária indexada no lote $\overline{ss}$ do tanque é então fixada nos valores 0 e 1 para não ser mais alterada. O próximo lote $\overline{ss} = \overline{ss} - 1$ passa a ser o índice das variáveis binárias no modelo relaxado. Novamente, as variáveis binárias dos lotes ss em t são recalculadas e fixadas.

Quando todos os lotes $\overline{ss}$ dos tanques em um macro-período forem fixados, a última configuração obtida para os lotes ss das linhas não será mais alterada. O algoritmo termina quando todos os lotes em todos os macro-períodos são percorridos ou caso o tempo de execução do método ultrapasse o limite estabelecido de 1 hora.

A idéia utilizada nesse algoritmo é fixar algumas variáveis como binárias e assim resolver um PPLR mais simples que o problema de programação inteira-mista estabelecido originalmente. O algoritmo forneceu melhores resultados quando percorremos os macro-períodos do fim para o início. Também foi testada a possibilidade de percorrer os macro-períodos no sentido contrário, mas isso aumentava o número de soluções infactíveis. Esse também foi o motivo pelo qual decidimos percorrer os lotes do fim para o início e por fixar primeiro um lote para tanques e depois os lotes para as linhas.

O método proposto aqui foi baseado nas idéias apresentadas em Clark and Clark (2000). O autor resolve um problema de dimensionamento de lotes em máquinas paralelas com tempos de preparo dependentes da seqüência. O modelo também é bastante complexo de ser resolvido e o autor propõe uma abordagem do tipo *fix & relax*.

A vantagem esperada nessa abordagem é o ganho no tempo computacional já que o número de variáveis binárias é reduzido consideravelmente. Todavia, devido a complexidade do nosso modelo matemático não temos garantia de que o algoritmo aqui proposto irá retornar uma solução factível.

A.2 Resultados computacionais

A tabela A.1 apresenta o número de soluções factíveis retornadas nas instâncias de pequena dimensão. O principal problema na método proposto foi a obtenção de soluções factíveis ao final de

$L/\overline{L}/J/\overline{J}$	$T=1$	$T=2$	$T=3$	$T=4$
2 / 2 / 2 / 1	10	9	10	10
2 / 2 / 3 / 2	8	5	6	4
2 / 2 / 4 / 2	6	1	5	3
3 / 2 / 2 / 1	10	10	10	10
3 / 2 / 3 / 2	8	7	4	7
3 / 2 / 4 / 2	7	9	4	8
4 / 2 / 2 / 1	10	10	10	7
4 / 2 / 3 / 2	7	9	5	5
4 / 2 / 4 / 2	7	6	7	10

Tab. A.1: Número de soluções factíveis obtidas

sua execução. A Heurística de Fixação de Variáveis não foi capaz de encontrar soluções factíveis para todas as instâncias dos conjuntos.

Iremos comparar a seguir a performance da HFV com os demais métodos. Todavia, deixamos claro que essa comparação está comprometida pelo fato da HFV não retornar o mesmo número de soluções factíveis que os demais métodos. As figuras A.1 e A.2 comparam o desempenho das soluções encontradas pelo HFV, AG e AM com a melhor solução encontrada pelo GAMS/Cplex.

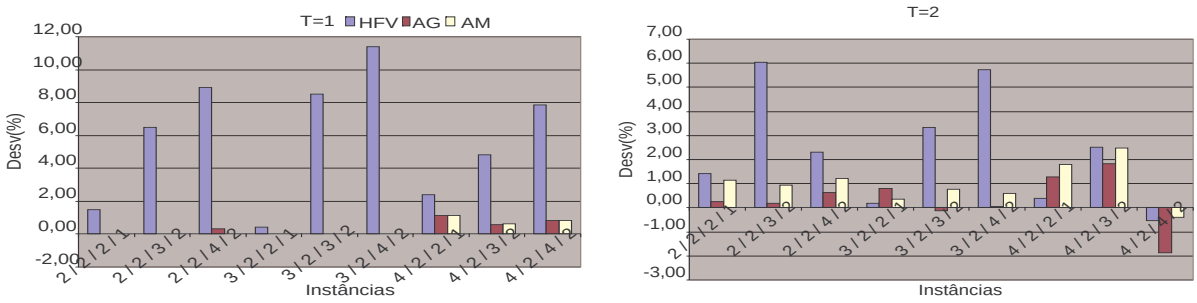

Fig. A.1: Comparação das heurísticas quando $T = 1$ e $T = 2$

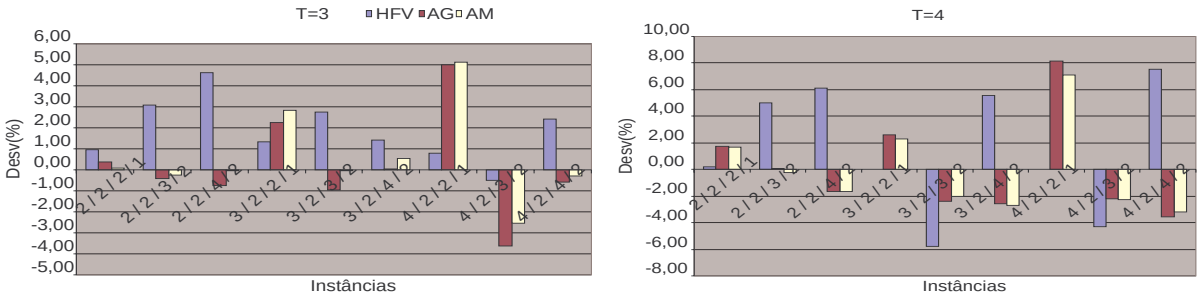

Fig. A.2: Comparação das heurísticas quando $T = 3$ e $T = 4$

O Desv considerado nas figuras anteriores foi o Desv Méd. As meta-heurísticas apresentaram

desempenho melhor que a heurística de fixação de variáveis na grande maioria dos conjuntos de instâncias. Apenas em algumas instâncias com $L = 2$ e $\overline{L} = 1$ a HFV apresenta desepenho melhor que as meta-heurísticas, ou seja, instâncias mais simples em termos de variáveis binárias envolvidas.

A única vantagem dessa abordagem está na redução considerável no tempo computacional, onde as soluções factíveis foram alcançadas. A tabela A.2 compara o tempo computacional médio do GAMS/Cplex e da HFV gasto para solucionar as instâncias de cada conjunto. Observamos que onde as soluções foram determinadas, a HFV levou, no pior caso, menos de 5 minutos para retornar uma solução. Ao contrário, o GAMS/Cplex já começa a levar mais de meia hora para retornar uma solução nos conjuntos onde $T = 2$.

	T=1		T=2		T=3		T=4	
$L/\overline{L}/J/\overline{J}$	GAMS	HFV	GAMS	HFV	GAMS	HFV	GAMS	HFV
2 / 2 / 2 / 1	2,08	0,02	28,27	1,37	3146,80	2,16	3377,44	5,56
2 / 2 / 3 / 2	0,96	0,03	1703,08	3,20	3076,89	6,09	3244,53	6,77
2 / 2 / 4 / 2	0,77	0,62	1545,06	3,64	2911,83	11,23	3600,10	12,54
3 / 2 / 2 / 1	573,32	1,21	3365,86	3,33	3600,06	6,40	3600,08	14,98
3 / 2 / 3 / 2	151,79	1,32	3600,08	8,26	3600,11	32,55	3600,12	20,26
3 / 2 / 4 / 2	46,26	1,36	3260,23	9,23	3600,11	12,75	3600,16	27,45
4 / 2 / 2 / 1	3068,64	1,69	3600,07	11,61	3281,30	13,36	3600,10	29,09
4 / 2 / 3 / 2	989,39	2,32	3254,55	14,23	3600,17	147,24	3600,16	56,71
4 / 2 / 4 / 2	1356,72	1,62	3600,11	14,16	3600,15	39,37	3600,19	226,27

Tab. A.2: Comparação do tempo computacional da HFV e GAMS/Cplex

A.3 Conclusão

A heurística de fixação de variáveis não conseguiu resultados relevantes nas instâncias de pequena dimensão. Alguns testes foram realizados em instâncias de elevada dimensão, mas o método não conseguiu determinar soluções factíveis na grande maioria dos conjuntos testados. Por isso, os resultados não foram apresentados aqui.

A tomada de decisão através da fixação das variáveis binárias parece não acompanhar a dinâmica envolvida no problema. Ao fixarmos o valor das variáveis binárias relativas a determinado lote, tomamos uma decisão que não poderá ser alterada posteriormente. Essa abordagem parece não ser adequada em um problema que estabelece uma sincronia entre cenários. Um procedimento que revisasse as decisões tomadas poderia ajudar a encontrar soluções factíveis. Isso nos levaria a uma abordagem próxima a um *branch & bound* e, conseqüêntemente, a um aumento no tempo computacional.

A força do método está na sua capacidade de reduzir o número de variáveis binárias provocando uma redução consideravel no tempo computacional. Um caminho para futuras pesquisas está na

possibilidade de encontrar um meio termo que permita revisar decisões já tomadas sem comprometer a rapidez do método.