UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA DE SISTEMAS

Planejamento da Produção de Múltiplos Itens com Restrições de Capacidade Através da Decomposição Cruzada

Kleber Xavier Sampaio de Souza

Prof. Dr. Vinícius Amaral Armentano
Orientador

Tese apresentada à Faculdade de Engenharia Elétrica da Universidade Estadual de Campinas - UNICAMP como parte dos requisitos exigidos para a obtenção do título de MESTRE EM ENGENHARIA ELÉTRICA.

23 de Agosto de 1989

Este exemplar corresponde à redação final da tese defendida por Kleber X. Sampaio de Souza e aprovada pela Comissão Julgadora em Setembro 1983

Vinícius A. Armentano

AGRADECIMENTOS

- A meus pais, sem os quais a realização deste trabalho seria impossível;
- Ao Prof. Dr. Vinícius Amaral Armentano pela orientação durante o desenvolvimento desta tese e estímulo à pesquisa;
- Aos professores do Departamento de Engenharia de Sistemas por seu apoio e dedicação;
- Aos professores José Cláudio Geromel, Paulo Morelato França e Roberto D. Galvão pela participação na banca examinadora;
- A Marcos Carneiro pelo código do programa para a resolução do PFCM;
- Ao amigo Amir Said pelos valorosos comentários sobre este trabalho;
- A Henrique pela amizade e confecção dos desenhos;
- Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico pelo apoio financeiro;

Abstract

O planejamento da produção de múltiplos itens com restrições de capacidade consiste na determinação das quantidades a serem produzidas em diferentes períodos de tempo na presença de restrições nos recursos disponíveis. O modelo apresentado neste trabalho inclui um custo de preparação para que esta produção possa ocorrer. A produção dos itens em um dado período é sujeita a limitações nas quantidades de horas regulares e horas extras. Além disso, o nível de produção de qualquer item em um dado período também é limitado. Este problema é resolvido através do método de Decomposição Cruzada, o qual pode fornecer uma solução ótima ou próxima da otimalidade dependendo do tempo computacional disponível. Este método unifica em um único procedimento a Decomposição de Benders e a Relaxação Lagrangeana.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O problema de planejamento da Produção de Múltiplos itens com restrições de Capacidade[1] tem sido um dos principais tópicos de pesquisa em gerenciamento de produção. Em um ambiente típico de manufatura, a decisão de produzir um item em um dado período implica em preparação dos meios de produção, quais sejam pessoal e maquinário para a sua confecção. Tem-se, então, necessariamente, um tempo e um custo gastos em tal preparação[2]. Tomando-se a decisão de produzir e estocar para os meses posteriores, reduz-se o custo global de preparação, mas incorre-se em custo de estocagem. Além disso, a produção a cada período é limitada de acordo com os recursos disponíveis. Ainda faz parte do planejamento a decisão sobre a quantidade de horas regulares e/ou horas extras também limitadas, de forma a minimizar os gastos com pessoal. O problema, então resulta da constante negociação entre todos os fatores objetivando o atendimento da demanda a custo mínimo.

Quando não existem restrições de capacidade o problema pode ser resolvido para cada item produzido pelo algoritmo de programação dinâmica de Wagner-Whitin[1]. Se os tempos e custos de preparação podem ser ignorados, o planejamento, ainda que capacitado, pode ser resolvido à otimalidade através de programação linear. Contudo, conforme já comentamos, em ambientes práticos de manufatura estas restrições estão presentes na maioria dos casos, tornando-se imperioso que sejam construídos modelos que tratem adequadamente do problema.

Valores significativos de preparação sugerem uma formulação através de programação inteira mista, com a variáveis binárias representando a preparação de uma dada combinação item-período. Como problemas típicos requerem um planejamento de vários de itens em longos horizontes de tempo, o problema assume grandes proporções, fazendo com que na maioria dos casos seja resolvido através de procedimentos heurísticos que levam a soluções

[1] Multi-Item Lot Size Problem

[2] set-up cost and set-up time

sub-ótimas.

A *Teoria da Complexidade Computacional* possibilita uma medida mais formal da complexidade do problema: ainda que a limitação nas quantidades produzidas seja ignorada, foi comentado em [21] que o problema resultante é *NP-completo*. Quando tais quantidades são limitadas, a intratabilidade do problema se mostra ainda mais evidente, pois cada um dos problemas resultantes da relaxação Lagrangeana já é *NP-hard* [2].Conseqüentemente é bastante improvável que exista um algoritmo computacionalmente eficiente[3] para a sua resolução pois, caso contrário, tal algoritmo poderia ser utilizado na resolução de todos os problemas da classe Não-Polinomial. Naturalmente, estes são resultados de pior caso, pois um problema exemplo mediano pode ser resolvido com razoável rapidez por métodos enumerativos.

O método de Enumeração Implícita do tipo Desdobramento e Sondagem[4] aplicado ao problema se torna bastante difícil em situações práticas devido aos muitos desdobramentos necessários para a obtenção de uma solução ótima. Como resultado, tal método é utilizado quase que exclusivamente na obtenção de heurísticas para o problema [3]. A decomposição de Benders foi aplicada por Bahl [4] ao problema onde não exite limitação nas quantidades produzidas,mas não foi feita uma análise de sua eficiência computacional. A técnica de subgradiente conjugada a procedimentos heurísticos apresenta melhor desempenho em problemas grandes que em pequenos [21], quando aplicada a problemas com limitação na quantidade total de horas disponíveis, mas sem limitação nas quantidades produzidas. Uma outra aplicação de subgradientes foi feita em Thizy et al. [22], onde também incorporou-se um algoritmo de fluxo em redes para que se tivesse um bom indicador da qualidade da solução a cada iteração.

Outras referências relevantes ao estado-da-arte enquanto procedimentos heurísticos são [5,6,7,8] onde são feitos comentários sobre a intratabilidade deste tipo de problemas e da quase inexistência de algoritmos exatos para a comparação com tais heurísticas de modo a se ter uma idéia do grau de acuracidade das mesmas.

O objetivo desta pesquisa é propor um algoritmo exato de resolução baseado no método de Decomposição Cruzada[5] [9], aplicável a problemas de tamanho razoável, seja para a obtenção de soluções ótimas, seja para testes em novas heurísticas. Não obstante, o principal mérito do trabalho reside na sugestão de uma forma diferente de abordagem do problema utilizando uma técnica recém desenvolvida e que tem demonstrado bons resultados quando aplicada a problemas de Localização de Facilidades [10], também *NP-hard*.

Em linhas gerais, o Método de Decomposição Cruzada explora simultaneamente as estruturas primal e dual de um programa inteiro misto, englobando em um único procedimento a Decomposição de Benders e a Relaxação Lagrangeana. Sua principal característica é a

[3] Convergência polinomial
[4] Branch-and-Bound
[5] Cross Decomposition

utilização dos subproblemas destes métodos agindo como mestres relaxados um do outro e os problemas mestres em si são justapostos para garantir a convergência global. Porém, nem todo algoritmo que utiliza os subproblemas primal e dual pode ser considerado como Decomposição Cruzada: veja-se, por exemplo, a aplicação de subgradientes feita por Thizy et al. [22],onde a cada iteração um problema de fluxo em redes a custo mínimo é resolvido para avaliar a qualidade da solução, mas nenhuma outra informação além do valor ótimo primal é considerada pelo método de resolução do problema dual.

Um fator muito importante é que o método pode ser utilizado como heurística e truncado no momento em que se desejar, pois a cada iteração tem-se limitantes superior e inferior bem como uma melhor solução factível encontrada até aquele instante. Com os limitantes pode-se ter uma idéia da qualidade da solução. No Capítulo 2 faremos uma análise do modelo de planejamento à luz da Decomposição Cruzada.

Como conseqüência da aplicação do Método temos quatro problemas com características bastante diversificadas: os dois problemas mestres e os dois subproblemas primais e duais. Para resolvê-los é feita no capítulo 3 uma análise detalhada da estrutura de cada um dirigida para a sua implementação computacional.

No Capítulo 4 são feitos testes computacionais verificando a adequação do método ao problema em questão e análise comparativa dos resultados obtidos

Finalmente, no Capítulo 5, são apresentadas as conclusões e feitas algumas sugestões para pesquisas futuras

Capítulo 2

Análise do Modelo através da Decomposição Cruzada

2.1 Formulação

O dimensionamento do tamanho de lotes[1] no planejamento da produçao de vários itens consiste da determinação da quantidade de cada item que deve ser produzida a cada período de um horizonte de tempo finito de modo a satisfazer demandas conhecidas.Este dimensionamento deve ser feito de forma a minimizar os custos fixos e variáveis de produção, de estocagem e de uso de recursos limitados, quais sejam a própria capacidade de produção e a quantidade de horas regulares e de horas extras disponíveis.

Um modelo aqui proposto para tal planejamento é o que se segue:

$$\text{(P) Min}_{X,I,\lambda,R,F} \quad \sum_{i,t} \sigma_{it}\lambda_{it} + \sum_{i,t} a_{it} I_{it} + \sum_{i,t} c_{it} X_{it} + \sum_t b_t R_t + \sum_t q_t F_t$$

sujeito a

$$I_{i,t-1} + X_{it} - I_{it} = d_{it}, \qquad i = 1,\ldots,I;\ t = 1,\ldots,T \tag{2.1}$$

$$X_{it} - \lambda_{it} M_{it} \leq 0, \qquad i = 1,\ldots,I;\ t = 1,\ldots,T \tag{2.2}$$

$$\sum_i (\lambda_{it} s_i + X_{it} p_i) - R_t \leq F_t, \qquad t = 1,\ldots,T \tag{2.3}$$

$$R_t \leq l_t, \qquad t = 1,\ldots,T \tag{2.4}$$

$$F_t \leq m_t, \qquad t = 1,\ldots,T \tag{2.5}$$

$$X_{it}, I_{it}, R_t, F_t \geq 0, \qquad i = 1,\ldots,I;\ t = 1,\ldots,T \tag{2.6}$$

$$\lambda_{it} \in \{0,1\} \qquad i = 1,\ldots,I;\ t = 1,\ldots,T \tag{2.7}$$

[1] Lot sizing.

onde:

I conjunto de itens;

T conjunto de períodos de tempo;

$I_{i,t}$ nível de estoque do item i no período t;

$X_{i,t}$ = produção do item i no período t;

R_t = quantidade de horas regulares requeridas no período t;

E_t = quantidade de horas extras requeridas no período t;

$d_{i,t}$ = demanda do item i no período t;

$\sigma_{i,t}$ = custo de preparação do item i no período t;

$a_{i,t}$ custo unitário de estocagem do item i no período t;

$v_{i,t}$ custo unitário de produção do item i no período t;

b_t custo unitário por hora regular

q_t custo unitário por hora extra;

p_i tempo de processamento unitário do item i;

s_i tempo de preparação para a produção do item i;

l_t limite de horas regulares disponíveis no período t;

m_t limite de horas extras no período t;

$M_{i,t}$ limite de produção do item i no período t;

$$\lambda_{i,t} \begin{cases} 1 & \text{se o item } i \text{ é produzido no período } t, \\ 0 & \text{caso contrário.} \end{cases}$$

Os conjuntos de restrições em $i = 1$ e $t = T$ são interpretados como:

- As restrições 2.1 consistem de balanços de estoque para cada item em cada período;
- As restrições 2.2 estabelecem um limite superior nos níveis de produção de dado item e também implicam em que uma preparação deve ser feita para qualquer nível positivo de produção;

- As retrições 2.3 impõem um equilibrio entre a quantidade de horas disponíveis e a quantidade de horas utilizadas na produção em cada peíodo;
- As restrições 2.4 e 2.5 impõem limites nas capacidades de horas regulares e horas extras disponíveis;

2.1.1 A Escolha do Modelo

Objetivando a adoção de um modelo adequado ao problema, foi feita uma análise dos encontrados na literatura:

Gelders et. al [11] 1. Considera os custos de preparação, mas não considera tempos de preparação;

2. Possui limitação apenas na quantidade de horas disponíveis, possuindo capacidade de produção ilimitada;

Evans [3] Possui limitação na quantidade estocada e na quantidade produzida, mas não considera os tempos de preparação;

Newson [8] Considera custos de estocagem, de produção e de preparação bem como tempos de preparação, mas não possui limitação na quantidade produzida;

Bahl et. al [4] Os custos de estocagem e de utilização da mão de obra são levados em conta, bem como os tempos de preparação, porém sua capacidade de produção é também ilimitada e não estabelece em máximo de horas extras disponíveis.Este modelo é semelhante ao de Newson.

Dos modelos anteriores, o que mais se aproxima do utilizado é o encontrado em Bahl, pois permitindo horas regulares e horas extras, necessita apenas que sua capacidade de produção seja limitada e que os custos de preparação sejam considerados. Como comentado na introdução, a capacitação nas quantidades produzidas traz consigo uma elevação no nível de complexidade do problema. Isto ocorre porque o problema Lagrangeno não pode mais ser resolvido pelo algoritmo de programação dinâmica de Wagner-Whitin [1] que é de convergência polinomial. Ao invés disto, sua resolução será feita por algoritmos mais elaborados, pois ainda para o caso especial em que todas as demandas são iguais, os custos de estocagem são nulos e es funções podem ser interpretadas como côncavas, com limites arbitrários de capacidade ou convexas com custo de preparação unitários, mostrou-se [2] que o problema resultante é *NP-hard*.

Será apresentado na próxima seção uma descrição geral do método de Decomposição Cruzada e sua aplicação no modelo de planejamento de produção aqui proposto.

2.2 O Método de Decomposição Cruzada

O princípio de decomposição de um problema em sua forma primal ou dual tem sido uma das técnicas mais utilizadas no desenvolvimento de algoritmos eficientes para a resolução de problemas de programação inteira mista.

Quando se tem um conjunto de variáveis *complicantes*, como por exemplo as variáveis inteiras de um programa inteiro misto, a decomposição de Benders tem-se mostrado particularmente vantajosa por permitir que a otimização nesse conjunto seja feita em separado, de modo que se possa utilizar algoritmos especializados[2].Quando aplicada a um problema de progtramação inteira mista, tal decomposição separa-o em um problema mestre envolvendo apenas variáveis inteiras[3] e um subproblema nas variáveis contínuas.

Uma iteração padrão consiste de:

1. Para uma dada combinação das variáveis inteiras, o subproblema nas variáveis contínuas é resolvido e suas variáveis duais determinam o corte dual mais violado;

2. Este corte e adicionado ao problema mestre que é então resolvido determinando uma nova combinação de variáveis inteiras.

O método se torna vantajoso quando:

a) Os cortes de Benders são fortes e somente alguns poucos são necessários [13];

b) O problema nas variáveis contínuas pode ser resolvido facilmente.

Quando a *Complicação* está em um subconjunto das restrições, como por exemplo, um conjunto do restrições unindo os vários blocos de uma matriz bloco-diagonal, ao aplicar-se a relaxação Lagrangeana [16] ou decomposição dual isola-se a otimização nestas restrições separando o problema em vários outros mais fáceis de resolver. Como resultado da aplicação da relaxação Lagrangeana também se tem um problema mestre e um subproblema.

Cada iteração pode ser resumida como:

1. Resolver o subproblema para um dado conjunto das variáveis duais, determinando mais um corte para o problema mestre;

2. Resolver o problema mestre encontrando um novo conjunto de variáveis duais.

A vantagem de sua aplicação está ligada à facilidade relativa de solução do subproblema em relação ao problema original.

[2] Como, por exemplo, o algoritmo de Balas [27] que é especializado para variáveis binárias.

[3] Exceto por uma variável real.

Contudo, muitos problemas podem ser explorados tanto de uma forma quanto de outra: o problema de localização de facilidades[4], conforme já mencionamos, é um exemplo, pois o subproblema resultante da relaxação Lagrangeana é reduzido à resolução de vários *problemas da mochila*[5] contínuos, um para cada facilidade [16]. Além disso, como pode ser encontrado em Lasdon [17], os dois métodos são duais entre si para o caso linear. Foi voltado para tais problemas que Van Roy desenvolveu o método de Decomposição Cruzada publicado em 1983 [9].

Na realidade, o cerne do método em si está na demonstração de que os dois subproblemas resultantes são mestres relaxados um do outro. Isto significa que se pode chegar a uma solução ótima apenas iterando entre estes subproblemas (ping-pong).

Para melhor definição, os quatro problemas componentes da Decomposição Cruzada serão, doravante, chamados mestre primal (MP) e subproblema primal (SP) aos resultantes da decomposição de Benders e mestre dual (MD) e subproblema dual (SD) aos resultantes da relaxação Lagrangeana. Na Figura 2.1 é apresentado um diagrama do método em versão simplificada. Uma iteração padrão consiste de:

1. Resolver o subproblema primal para dados valores das variáveis inteiras, encontrando as variáveis duais correspondentes;

2. Para um subconjunto fixo das variáveis duais encontradas resolver o subproblema dual determinando novos valores para as variáveis inteiras.

Porém, o algoritmo composto apenas pelos dois subproblemas não tem convergência garantida. Para assegurar tal convergência,o ping-pong é imerso em um dos esquemas padrões de decomposição cuja convergência seja comprovada. Quando o progresso feito com o ping-pong cessa, as soluções previamente geradas são utilizadas na construção de cortes para o problema mestre primal ou dual, que é então resolvido determinando uma solução que servirá como ponto de partida na continuação do ping-pong entre os subproblemas. As principais vantagens na utilização dos dois problemas agindo problemas mestres relaxados um do outro são:

1. Os subproblemas, devido a suas estruturas especiais, são mais fáceis de resolver que os problemas mestres originais;

2. Sob determinadas circunstâncias, que serão mostradas posteriormente, é possível a convergência apenas com o ping-pong;

3. Também será mostrado que os limitantes dados pelos subproblemas são superiores aos dos próprios problemas mestres fazendo com que se tenha a cada iteração uma melhor visão sobre a qualidade da solução atual.

[4]Facility location problem.
[5]Knapsack problems.

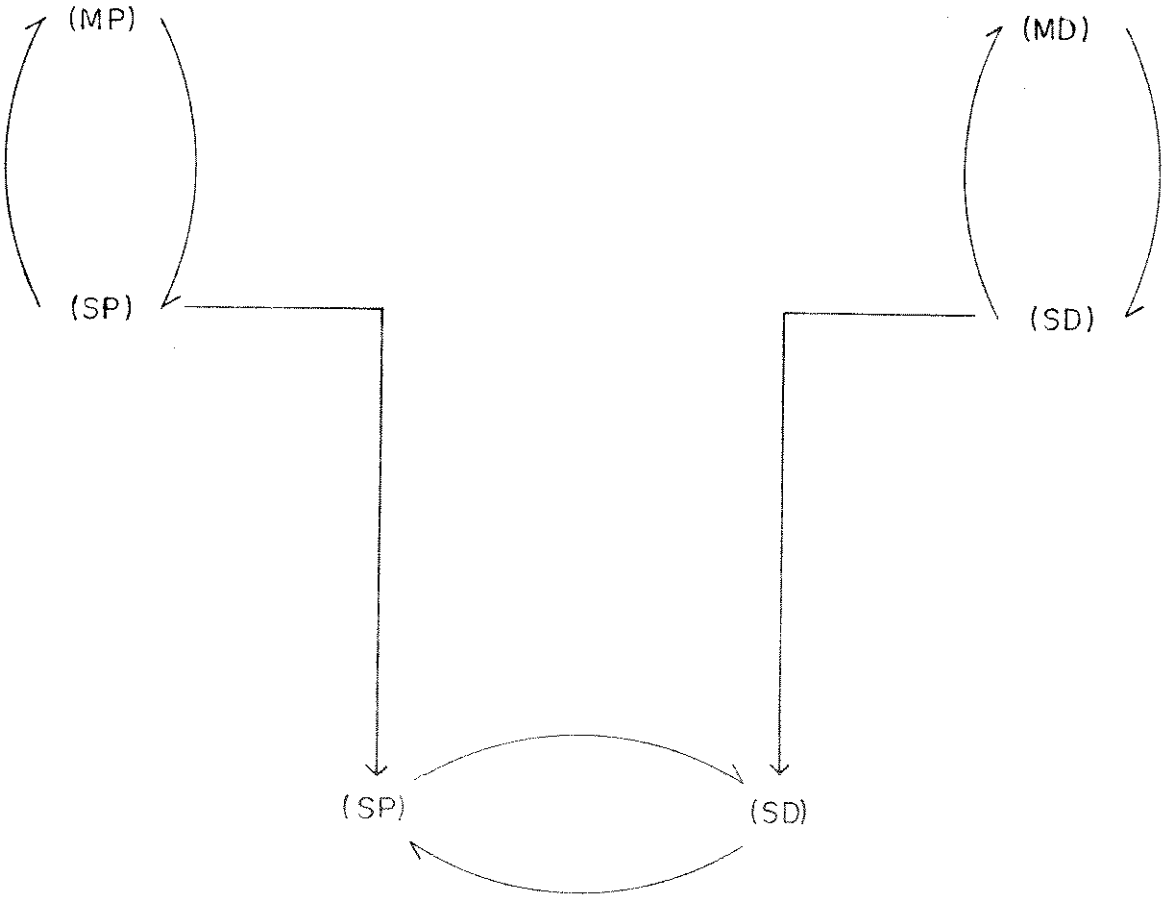

Figura 2.1.: Versão simplificada do método de decomposição cruzada.

O modo como os mestres são anexados para garantir a convergência global será retomado quando da construção formal do algoritmo. Far-se-á, agora, a construção do método em si, adaptando os teoremas apresentados em [9] ao modelo utilizado. Primeiramente será aplicada a decomposição de Benders e, em seguida a relaxação Lagrangeana, tendo-se como resultado os problemas mestres (MP) e (MD) e os subproblemas (SP) e (SD) conforme já comentado.

Para maior facilidade de exposição, a seguinte notação será utilizada: se $(\cdot)$ é um problema de otimização, então $v(\cdot)$ é o valor de sua solução ótima e $F(\cdot)$ sua região factível.

2.2.1 Decomposição de Benders

O subproblema da decomposição de Benders (SP_λ) é obtido fixando-se as variáveis binárias no problema (P) em zero ou em um:

$$(SP_\lambda)\ \text{Min}_{X,I,R,F} \quad \sum_{i,t} \sigma_{i,t}\lambda_{i,t} + \sum_{i,t} a_{i,t} I_{i,t} + \sum_{i,t} v_{i,t} X_{i,t} + \sum_t b_t R_t + \sum_t q_t F_t$$

sujeito a

$$\begin{aligned}
I_{i,t-1} + X_{i,t} - I_{i,t} &= d_{i,t}, && i \in I, t \in T && (2.8)\\
X_{i,t} &\leq \lambda_{i,t} M_{i,t}, && i \in I, t \in T && (2.9)\\
\sum_i X_{i,t} p_i - R_t - F_t &\leq \sum_i \lambda_{i,t} s_i, && t \in T && (2.10)\\
R_t &\leq l_t, && t \in T && (2.11)\\
F_t &\leq m_t, && t \in T && (2.12)\\
X_{i,t}, I_{i,t}, R_t, F_t &\geq 0, && i \in I, t \in T && (2.13)
\end{aligned}$$

cujo dual é dado por:

$$(DSP)\ \text{Max}_{A,B,C,D,E} \quad \sum_{i,t} d_{i,t} A_{i,t} + \sum_{i,t} M_{i,t}\lambda_{i,t} B_{i,t} + \sum_t \left[(\sum_i s_i \lambda_{i,t}) C_t + l_t D_t + m_t E_t \right]$$

sujeito a

$$\begin{aligned}
A_{i,t} + A_{i,t+1} &\leq a_{i,t}, && i \in I, t \in T && (2.14)\\
A_{i,T+1} &\geq 0, && i \in I && (2.15)\\
A_{i,t} - B_{i,t} + p_i C_t &\leq v_{i,t}, && i \in I, t \in T && (2.16)\\
-C_t + D_t &\leq b_t, && t \in T && (2.17)\\
C_t + E_t &\leq q_t, && t \in T && (2.18)\\
B_{i,t}, D_t, E_t &\geq 0, && i \in I, t \in T && (2.19)
\end{aligned}$$

onde $A_{i,t}, B_{i,t}, C_t, D_t$ e E_t são as variáveis duais correspondentes às restrições de 2.8 a 2.12, respectivamente

Será mostrado na seção 3.1.3 que $P(DSP)$ pode ser assumida como limitada. Deste modo, o problema mestre primal com todos os seus cortes, denotado por (MP_A), pode ser escrito como:

$$
\begin{aligned}
(MP_A)\ \text{Min}_{\lambda,\rho} \quad & \rho \\
\text{sujeito a} \quad & \\
\rho \geq & \sum_{i,t} d_{i,t} A^j_{i,t} + \sum_t l_t D^j_t + \sum_t m_t E^j_t \\
& + \sum_{i,t} (\sigma_{i,t} + M_{i,t} B^j_{i,t} - C^j_t s_t) \lambda_{i,t}, j \in J_{PA} && (2.20) \\
& \lambda_{i,t} \in \{0,1\} && (2.21) \\
& \rho \text{ irrestrito} && (2.22)
\end{aligned}
$$

onde J_{PA} é o conjunto de índices de todos os pontos extremos do politopo de (DSP).

As restrições indexadas por J_{PA} são chamadas cortes de Benders e seu número, normalmente, é muito grande. Então, Benders propôs resolver uma relaxação (MP) de (MP_A) tomando um pequeno número de cortes e a geração de outros quando necessário.

2.2.2 Relaxação Lagrangeana

Originalmente, ao se representar matricialmente o problema (P), observa-se uma estrutura bloco angular da seguinte forma:

$$
\begin{bmatrix}
\mathbf{L_1} & \mathbf{L_2} & \cdots & \mathbf{L_N} \\
\mathbf{A_1} & & & \\
 & \mathbf{A_2} & & \\
 & & \ddots & \\
 & & & \mathbf{A_N}
\end{bmatrix} \qquad (2.23)
$$

Isto, então, pode ser visto como um problema de minimização do custo total de N programas distintos que são independentes a menos de um subconjunto das restrições que os está ligando.A envoltória convexa de cada uma das regiões factíveis destes problemas é limitada [6] fazendo com que constituam um conjunto de poliedros. Além disso, os programas correspondentes a cada bloco já foram extensivamente explorados em sua forma de resolução [1,18,19].

[6] Pois cada variável é limitada.

Tudo isto indica claramente que a melhor forma de relaxação é através das restrições de ligação, resultando no subproblema:

$$(SD_C)\ \text{Min}_{X,I,R,F,\lambda} \quad \sum_{i,t}(\sigma_{i,t} - s_i C_t)\lambda_{i,t} + \sum_{i,t} I_{i,t} a_{i,t} + \sum_{i,t}(v_{i,t}/p_i - C_t)p_i X_{i,t} + \\ + \sum_t (b_t + C_t) R_t + \sum_t (q_t + C_t) F_t$$

sujeito a

$$\begin{aligned} I_{i,t-1} + X_{i,t} - I_{i,t} &= d_{i,t}, && i \in I, t \in T \\ X_{i,t} &\leq \lambda_{i,t} M_{i,t}, && i \in I, t \in T \\ R_t &\leq l_t, && t \in T \\ F_t &\leq m_t, && t \in T \\ X_{i,t}, I_{i,t}, R_t, F_t &\geq 0, && i \in I, t \in T \\ \lambda_{i,t} &\in \{0,1\} && i \in I, t \in T \end{aligned}$$

Expresso em termos de planos de corte o problema mestre pode ser escrito como:

$$(MD_A)\ \text{Max}_{\delta,C} \quad \delta$$

sujeito a

$$\delta \leq \sum_{i,t} \sigma_{i,t}\lambda^j_{i,t} + \sum_{i,t} a_{i,t} I^j_{i,t} + \sum_t b_t R^j_t + \sum_t q_t F^j_t + \sum_{i,t} v_{i,t} X_{i,t} + \\ + \sum_t [F^j_t + R^j_t - \sum_i (X^j_{i,t} p_i + \lambda^j_{i,t} s_i)]C_t, j \in J_{DA} \qquad (2.24)$$

$$\delta, C_t \text{ irrestritos} \qquad (2.25)$$

onde J_{DA} é o conjunto de todos os pontos da envoltória convexa de $F(SD_C)$.

Novamente, tem-se um número de cortes muito grande e o que se propõe no método de geração de planos de cortes é resolver, como na Decomposição de Benders, o problema mestre relaxado, gerando os cortes quando necessário.

De modo a estabelecer as relações de equivalências entre os problemas envolvidos nos dois métodos é necessária a definição de alguns conjuntos:

(MP) $\{MP : j \in J_P \subset J_{PA}\}$;

J_C $\{j \in J_{PA} : (A^j_{i,t}, B^j_{i,t}, C^j_t, D^j_t, E^j_t)$ com $C^j_t = C_t$ definindo um corte gerado por um ponto extremo de $(DSP)\}$;

J_C $\{j \in J_{PA} : (A^j_{i,t}, B^j_{i,t}, C^j_t, D^j_t, E^j_t)$ definindo um corte gerado por um ponto extremo do dual de (SP_λ) quando C^j_t é fixado *a priori* $\}$;

$(MP_C) = \{MP : j \in J_C\}$.

A definição de (MP) é, na realidade, a formalização do problema mestre de Benders em sua forma relaxada. J_C representa um subconjunto de (MP_A) que contém todos os cortes para um C fixo, enquanto $\bar{J}_C$ é o conjunto de cortes que seriam gerados se C_t tivesse sido fixado quando da resolução de (DSP). Entre os conjuntos J_C e $\bar{J}_C$ existe uma dirença muito sutil, pois caso o politopo dual tenha a forma da Figura 2.2.2 o problema mestre completo (MP_A) tem apenas três valores distintos de C_t em seus cortes a saber, os pontos 1,2 e 3. Então, caso se fixasse o valor de C_t^1, por exemplo, J_C e $\bar{J}_C$ seriam diferentes: $\bar{J}_C = \{2\}$ e $J_C = \{2,4\}$, pois com C_t^+ fixado na resolução de (DSP) existiriam dois cortes possíveis para C_t^+ quais sejam, o 2 e o 4. Portanto J_C é um subconjunto de J_{PA} enquanto $\bar{J}_C$ não o é em geral.

É fácil mostrar [9] que as seguintes relações se verificam:

$$J_C \subset J_{PA}$$

$$J_C \subseteq \bar{J}_C$$

e além disso,

$$J_C \subseteq J_{PA} \Rightarrow J_C = \bar{J}_C$$

Os cortes pertencentes à diferença $\bar{J}_C \setminus J_C$ não fazem parte do problema mestre de Benders original. Porém, desde que a região $F(DSP)$ é assumida como limitada, qualquer ponto não extremo $(A_{i,t}^k, B_{i,t}^k, C_t^k, D_t^k, E_t^k)$ pode ser escrito como combinação convexa de pontos extremos desta mesma região:

$$(A_{i,t}^k, B_{i,t}^k, C_t^k, D_t^k, E_t^k) = \sum_{j \in K} \alpha_j (A_{i,t}^j, B_{i,t}^j, C_t^j, D_t^j, E_t^j)$$

onde:

$$K \subset J_{PA}, \sum_{j \in K} \alpha_j = 1.$$

Um corte de Benders gerado por este ponto não extremo é dado por:

$$\rho \geq \sum_{i,t} d_{i,t} A_{i,t}^k + \sum_t l_t D_t^k + \sum_t E_t^k + \sum_{i,t} (\sigma_{i,t} + M_{i,t} B_{i,t}^k - C_t^k s_i) \lambda_{i,t}$$

Substituindo as variáveis por suas combina ções convexas, tem-se:

$$\rho \geq \sum_{i,t} d_{i,t} \sum_{j \in K} \alpha_j A_{i,t}^j + \sum_t l_t \sum_{j \in K} \alpha_j D_t^j + \sum_t m_t \sum_{j \in K} \alpha_j E_t^j + \sum_{i,t} (\sigma_{i,t} + M_{i,t} \sum_{j \in K} \alpha_j B_{i,t}^j - s_i \sum_{j \in K} \alpha_j C_t^j) \lambda_{i,t}$$

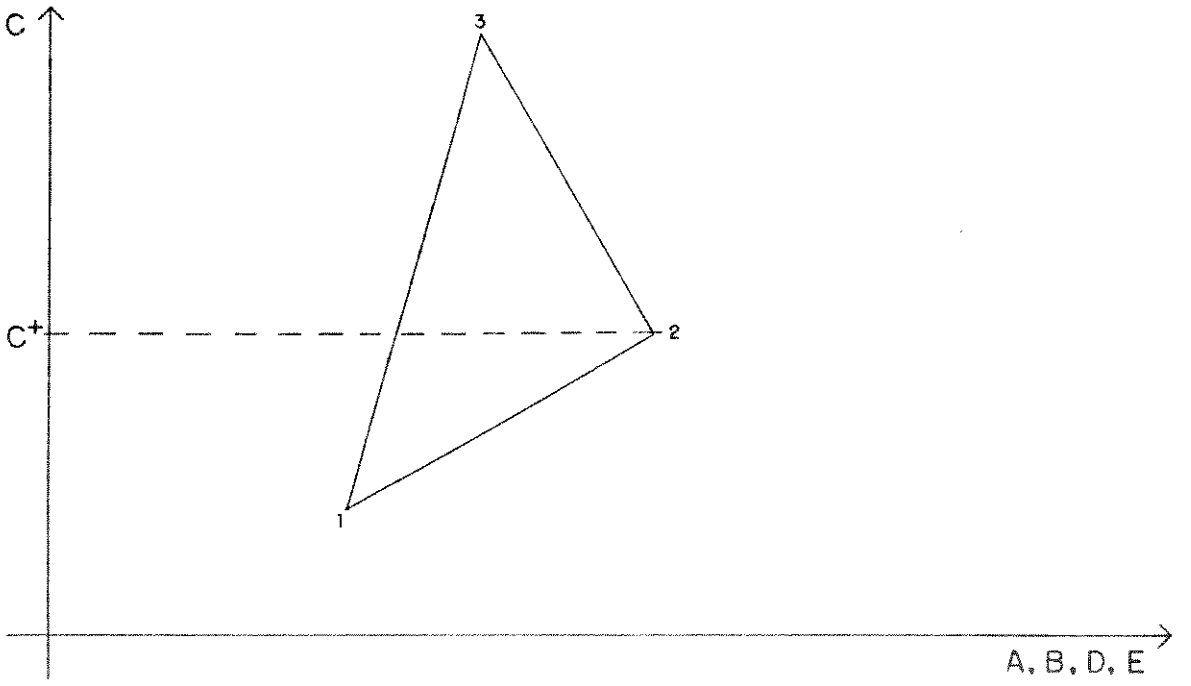

Figura 2.2 : Exemplo de politopo dual.

desde que $\sum_{j \in K} \alpha_j = 1$:

$$\sigma_{i,t} \sum_{j \in K} \alpha_j \geq \sigma_{i,t}$$

e

$$\rho \sum_{j \in K} \alpha_j \geq \rho$$

então o corte pode ser escrito na forma:

$$\sum_{j \in K} \alpha_j \rho \geq \sum_{j \in K} \alpha_j [\sum_{i,t} d_{i,t} A^j_{i,t} + \sum_t l_t D^j_t + \sum_t E^j_t + \sum_{i,t} (\sigma_{i,t} + M_{i,t} B^j_{i,t} - C^j_t s_i) \lambda_{i,t}] \quad (2.26)$$

que representa uma combinação convexa dos cortes gerados pelos pontos extremos com índices $j \in J_C \setminus J_C$ podem ser escritos como uma combinação dos cortes com $j \in K \subset J_{PA}$ e são, portanto, redundantes em (MP_A).

Notando-se que $(MP_C) = \{MP : j \in J_C\}$ pode ser escrito como $\{MP : j \in J_C \cup j \in J_C \setminus J_C\}$ e fazendo-se $\rho_A = v(MP_A)$ tem-se que

$$\rho_A \geq \sum_{i,t} d_{i,t} A^j_{i,t} + \sum_t l_t D^j_t + \sum_t E^j_t + \sum_{i,t} (\sigma_{i,t} + M_{i,t} B^j_{i,t} - C^j_t s_i) \lambda_{i,t}, j \in J_C$$

porque $J_C \subset J_{PA}$. Substituindo-se $\rho = \rho_A$ na desigualdade 2.26 verifica-se que a condição também é válida para os cortes com $j \in J_C \setminus J_C$, concluindo-se que:

$$v(MP_A) \geq v(MP_C).$$

Além disto, como $J_C \subset J_C$

$$v(MP_C) \geq v(MP : j \in J_C)$$

tendo-se, finalmente,

$$v(MP_A) \geq v(MP_C) \geq v(MP : j \in J_C) \quad (2.27)$$

Definição 2.1 *Um problema (P) é equivalente a um outro (Q) com respeito a um subconjunto de suas variáveis, se uma solução ótima neste subconjunto para um deles for ótima também para o outro.*

O Teorema 2.1 a seguir está provado em [9]:

Teorema 2.1 *A relaxação do problema mestre de Benders (MP_C) é equivalente ao subproblema da relaxação Lagrangeana (SD_C) com respeito às variáveis primais $\lambda_{i,t}$ e $v(MP_C) = v(SD_C)$.*

É importante notar que esta relação de equivalência indica que se pode usar o subproblema dual (SD_C) como mestre relaxado do subproblema primal (SP_λ) com as seguintes vantagens:

1. O subproblema (SD_C) tem sua estrutura explorada pelo próprio fato de ser resultante da relaxação Lagrangeana, onde se procura dar uma melhor tratabilidade ao problema atráves da dualização de restrições *complicantes*. Como já foi comentado, no modelo utilizado a dualização separa os vários blocos de uma matriz bloco-angular e os vários problemas resultantes podem ser resolvidos ao mesmo tempo utilizando computação paralela, enquanto que o problema mestre original é sabido ser o *calcanhar de Aquiles* do método de Benders por sua dificuldade de resolução.

2. O subproblema (SD_C) considera todos os cortes presentes em $J_{C'}$, que é uma situação ainda melhor que aquela na qual todos os cortes em $J_{C'}$ são considerados. Daí e das desigualdades 2.27

$$v(MP_A) \geq v(MP_{C'}) = v(SD_{C'}) \geq v(MP : j \in J_{C'})$$

   É importante notar que todos esses cortes são considerados implicitamente, isto é, nenhum deles é gerado.

Subconjuntos similares de cortes e problemas relaxados podem ser definidos a partir de (MD) e J_{DA}:

- Exatamente como na decomposição de Benders o problema mestre é relaxado e os cortes são gerados à medida que vão sendo requisitados pelo algoritmo. O problema relaxado é escrito como

$$(MD) \quad \{MD : j \in J_D \subset J_{DA}\}$$

- Os cortes do mestre dual completo (MD_A) com o mesmo valor de $\lambda_{i,l}$ são agrupados formando o conjunto:

$$J_\lambda = \{j \in J_{DA} : (\lambda^j_{i,l}, X^j_{i,l}, I^j_{i,l}, R^j_l, F^j_l) \text{ com } \lambda^j_{i,l} \quad \lambda_{i,l} \text{ define um corte gerado por um ponto extremo de } (SD_{C'})\}$$

- Novamente, um conjunto de cortes que seriam redundantes ao mestre completo é considerado para, possivelmente, melhorar o desempenho do problema mestre que utiliza somente aqueles com $\lambda_{i,l}$ fixo:

$$J_\lambda \quad \{j : (\lambda^j_{i,l}, X^j_{i,l}, I^j_{i,l}, R^j_l, F^j_l) \text{ define um corte gerado por um ponto extremo de } (SD_{C'}) \text{ quando } \lambda_{i,l} \text{ é fixado } \textit{a priori}\}$$

- A partir do conjunto J_λ define-se um problema mestre relaxado (MD_λ)

$$(MD_\lambda) \quad \{MD : j \in J_\lambda\}$$

Analogamente às conclusões sobre os conjuntos J_ℓ, J_C e ao problema (MP_ℓ) pode-se deduzir que:

$$J_\lambda \subset J_\lambda$$

$$J_\lambda \subset J_{DA}$$

$$J_\lambda \subseteq J_{DA} \Rightarrow J_\lambda = J_\lambda$$

e que:

$$v(MD : j \in J_\lambda) \geq v(MD_\lambda) \geq v(MD_A) \tag{2.28}$$

Note das desigualdades 2.28 que o problema (MD_λ) é superior a $(MD : j \in J_\lambda)$.

O Teorema 2.2 a seguir, provado em [9], mostra que se pode usar o subproblema primal como mestre relaxado do subproblema dual, completando o que se chama ping-pong entre os subproblemas primal e dual. Sua vantagem de utilização é a exploração da estrutura do subproblema primal, que geralmente permite uma melhor tratabilidade do que o problema mestre dual relaxado (MD_λ).

Teorema 2.2 *A relaxação (MD_λ) do problema mestre dual é equivalente ao subproblema da decomposição de Benders (SP_λ) com respeito à variável dual C_ℓ e $v(MD_\lambda) = v(SP_\lambda)$.*

Será mostrado que (SP_λ) pode ser resolvido por um algoritmo de fluxo em redes, enquanto (MD_λ), não possuido nenhuma estrutura especial, deve ser resolvido por um programa linear genérico.

A Figura 2.3 mostra o ping-pong entre os subproblemas primal e dual. Antes de montar o algoritmo completo deve-se estabelecer um critério de convergência que será mostrado na seção seguinte.

2.2.3 Convergência

As iterações entre os subproblemas (SP_λ) e (SD_C) apresentam uma perda de informação por serem estes problemas mestres *relaxados* um do outro. Deve-se ver que embora considerando cortes gerados por soluções não básicas, o que os torna mais restritos que $(MD : J \in \overline{J}_\lambda)$ e $(MP : j \in J_\ell)$, eles ainda são uma relaxação e, como tal, podem perder alguma informação crucial à convergência global. Quando esta perda realmente acontece o ping-pong cessa o seu progresso na melhoria dos incumbentes e imerge em um processo de ciclagem. Caso contrário, na ausência de gap de dualidade, o algoritmo pode convergir apenas deste modo.

De modo a assegurar a convergência em qualquer caso vejamos o Lema 2.1 provado em [9].

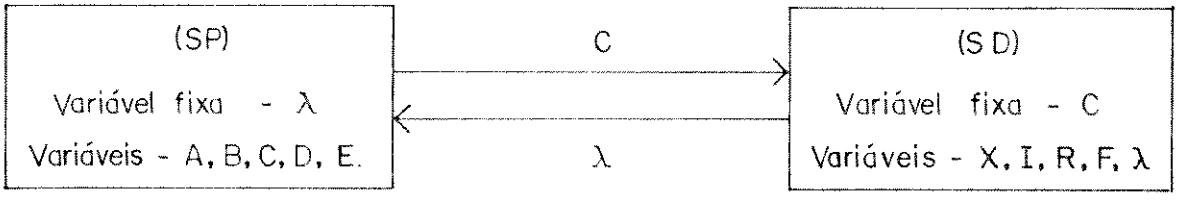

Figura 2.3.: Ping-pong entre os subproblemas primal e dual.

Lema 2.1 **(a)** *Seja* $(A^v_{i,t}, B^v_{i,t}, C^v_t, D^v_t, E^v_t)$ *uma solução dual ótima de* (SP_{λ^v}) *e* $(X^+_{i,t}, I^+_{i,t}, R^+_t, F^+_t, \lambda^+_{i,t})$ *uma solução ótima de* (SD_{C^v}). *Então, a menos que* $v(SP_{\lambda^v}) = v(P)$, $\lambda^+_{i,t} \neq \lambda^v_{i,t}$.

(b) *Seja* $(X^v_{i,t}, I^v_{i,t}, R^v_t, F^v_t, \lambda^v_{i,t})$ *uma solução ótima de* (SD_{C^v}) *e* $(A^+_{i,t}, B^+_{i,t}, C^+_t, D^+_t, E^+_t)$ *uma solução dual ótima de* (SP_{λ^v}). *Então, a menos que* $v(SD_{C^v}) = v(P), C^+_t \neq C^v_t$.

O Lema 2.1 garante que não pode haver replicação em um número de iterações menor que quatro a menos que o problema esteja resolvido[7].

$$\xrightarrow{\lambda^0} (SP_{\lambda_0}) \xrightarrow{C^0} (SD_{C^0}) \xrightarrow{\lambda^1} (SP_{\lambda_1}) \xrightarrow{C^1} (SD_{C^1}) \xrightarrow{\lambda^2} \tag{2.29}$$

Observando-se a sequência 2.29, a menos que a solução tenha sido encontrada, $\lambda^1 \neq \lambda^0$, $\lambda^2 \neq \lambda_1$ e $C^0 \neq C^1$, mas pode ocorrer $\lambda^0 = \lambda^2$ pois o lema só se aplica a soluções consecutivas. Daí, pode ocorrer replicação a partir da quarta iteração fechando um ciclo[8]. A solução encontrada para estabelecer um teste de convergência foi a contagem do número de iterações sem a melhoria dos incumbentes. Quando este número chegar a quatro, deve-se resolver um dos problemas mestres para restaurar a informação perdida e restabelecer o ping-pong continuando o processo.

A justaposição de um problema mestre ao ping-pong representa, na verdade, a inclusão deste num procedimento finito para que sua convergência seja garantida.

Embora a inclusão de apenas um dos problemas mestres já garanta a convergência[9], pode-se adicionar os dois, objetivando a uma recuperação da informação no problema mestre correspondente ao que está gerando a falha. Por exemplo: no diagrama anterior, a falha foi detectada logo após o subproblema dual, logo, este tornou-se incapaz de substituir o problema mestre primal, que é resolvido a seguir.

A forma geral do método de decomposição cruzada é apresentada na Figura 2.4. A partir desta forma geral podem ser derivadas algumas variações sem que a convergência seja prejudicada:

- O problema mestre primal (MP) pode ser retirado desde que seja utilizado um algoritmo de desdobramento e sondagem para fechar um eventual gap de dualidade;
- O problema mestre dual pode ser retirado, pois a decomposição de Benders, composta por (MP) e (SP_λ), tem convergência garantida para a solução primal;
- Poder-se fazer a maximização da função dual através de subgradientes ao invés de se resolver (MD), contanto que (MP) seja mantido. A principal vantagem desta aborda-

[7] Considerando que cada subproblema resolvido representa uma iteração.

[8] Uma conclusão semelhante pode ser obtida a partir de uma sequência iniciando com C^0.

[9] Do problema dual ou do primal, conforme o problema mestre incluído.

gem é que os multiplicadores duais C_t são calculados com menor esforço computacional em relação à utilização de um algoritmo de programação linear.

Note que quando um dos problemas mestres não é utilizado a geração de seus cortes é, obviamente, desnecessária.

Das opções acima, optou-se pela terceira com o retorno ao ping-pong após um certo número de iterações de subgradiente fixado heuristicamente, com o objetivo da obtenção de multiplicadores C_t de melhor qualidade e,conseqüentemente com maior probabilidade de restabelecer com sucesso o ping-pong.

O Teorema 2.3, a seguir, mostra que a otimalidade pode ser verificada em uma única iteração do método de Decomposição Cruzada (Van Roy [9]).

Teorema 2.3 *Seja* $(X_{i,t}, I_{i,t}, R_t, F_t, \lambda_{i,t})$ *uma solução ótima de* (P) *e* $(A_{i,t}, B_{i,t}, C_t, D_t, E_t)$ *uma solução ótima de* (D). *Então as afirmações seguintes são equivalentes:*

(a) *A relaxação Lagrangeana relativa às restrições 2.8 não possui gap de dualidade, isto é,* $v(P) = v(D)$;

(b) *Existe uma solução dual ótima* $(A^1_{i,t}, B^1_{i,t}, C^1_t, D^1_t, E^1_t)$ *de* (SP_{λ^*}) *com* $v(SD_{C^*}) = v(SP_{\lambda^*})$;

(c) *Existe uma solução primal ótima* $(X^1_{i,t}, I^1_{i,t}, R^1_t, F^1_t, \lambda^1_{i,t})$ *de* (SD_{C^*}) *com* $v(SP_{\lambda^*}) = v(SD_{C^*})$

O Teorema 2.3 indica que após encontrada as soluções ótimas[10] de um dos problemas, $v(P) = v(D)$ implica que existe uma delas que satisfaz $v(SD_{C^*}) = v(D) = v(P) = v(SP_\lambda)$, finalizando o algoritmo já na próxima iteração. Infelizmente, porém, não são todas as soluções que verificam a igualdade, devendo-se escolher cuidadosamente a solução a ser utilizada na próxima iteração. No entanto, se a solução é única ou a parte dela a ser utilizada na próxima iteração assim o for, como por exemplo, a solução é múltipla com todos os C_t iguais, a igualdade será verificada. No modelo em estudo, a existência de multiplicidade no vetor C_t é estudada no Capítulo 3, objetivando à escolha da solução a ser utilizada no próximo passo.

Com as considerações acerca da convergência, conclui-se o algoritmo de decomposição cruzada que é apresentado a seguir:

1. INICIALIZAÇÃO

   - $C_t = 0, t = T$;
   - $vp = \infty$;

[10] Em caso de soluções múltiplas.

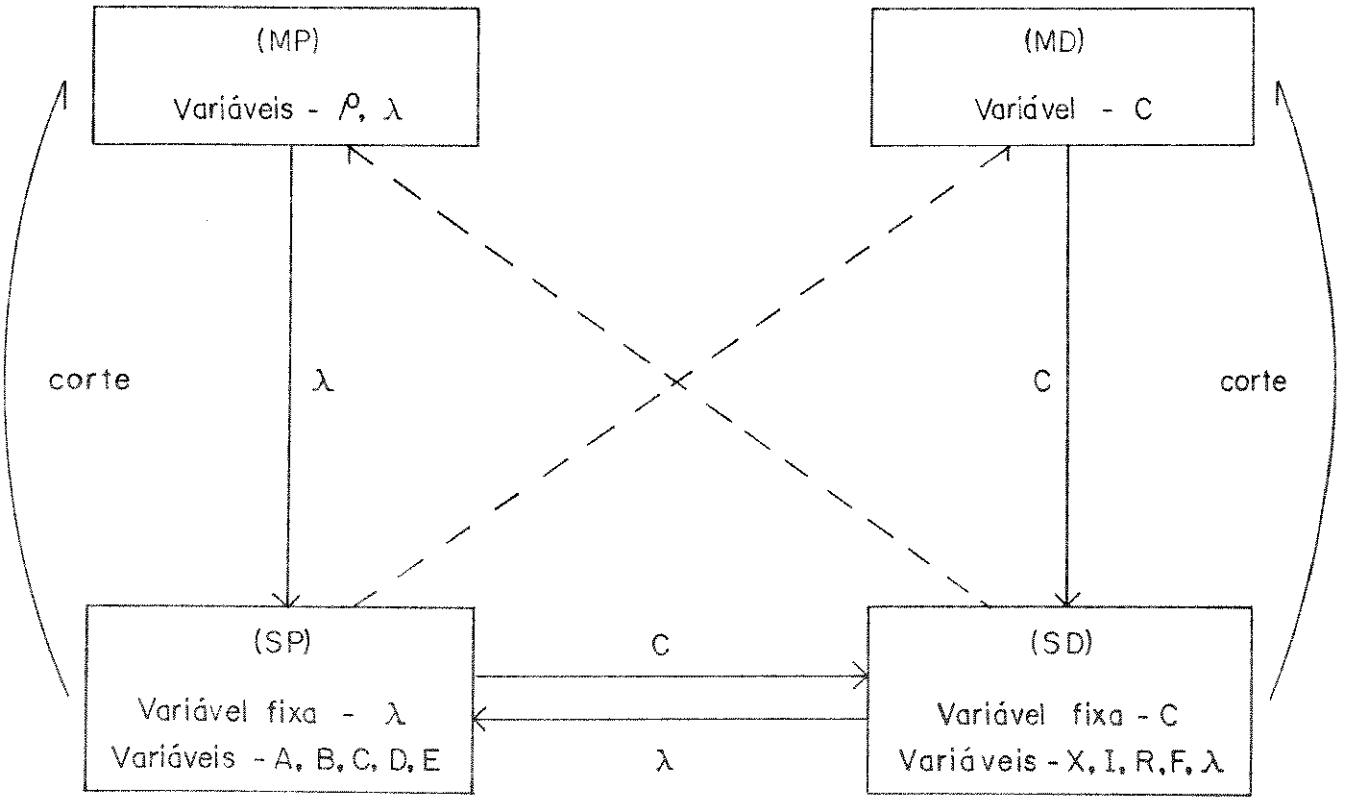

Figura 2.4.: Diagrama do método de decomposição cruzada.

- $\bar{v}_D = v(P) = -\infty$;
- τ = TOLERÂNCIA;
- G = GAP PARA UTILIZAÇÃO DE SUBGRADIENTE;

2. ENQUANTO $v_D(1+G) < v_P$: o subgradiente ainda pode ser vantajoso

   (a) ENQUANTO houver convergência (cont<4):

      i. RESOLVER o subproblema dual;

      ii. **SE** $v_D < v(SD)$, houve melhoria do incumbente dual

         ATUALIZAR o incumbente dual: $v_D = v(SD)$;

         REINICIALIZAR o contador de convergência: cont = 1;

         CASO CONTRÁRIO incrementar o contador de convergência: cont=cont+1;

      iii. SE $v_D(1+\tau) \geq v_P$, a solução ótima foi encontrada e é a correspondente ao último (SP) resolvido. FIM.

      iv. SE cont=4 cessou o progresso no ping-pong:

         RESOLVER problema mestre primal;

         ATUALIZAR $v(P) = v(MP)$;

      v. SE $\bar{v}_P \leq (1+\tau)v(P)$, a solução ótima foi encontrada e é a correspondente ao último (SP) resolvido. FIM.

      vi. RESOLVER o subproblema primal;

      vii. GERAR corte de Benders;

      viii. **SE** $v_P > v(SP)$, houve melhoria do incumbente primal

         ATUALIZAR o incumbente primal : $v_P = v(SP)$;

         REINICIALIZAR o contador de convergência: cont = 1;

         CASO CONTRÁRIO incrementar o contador de convergência: cont = cont + 1;

      ix. SE $v_D(1+\tau) > v_P$ ou $v_D > (1+\tau)v(P)$: ótimo encontrado. FIM.

      x. SE $v_D(1+G) > v_P$ FECHAR GAP;

   (b) RESOLVER SUBGRADIENTE;

   (c) REINICIALIZAR o contador de convergência: cont = 1;

Para fechar gap de dualidade pode-se proceder com um algoritmo de *Desdobramento e Sondagem* ou com a Decomposição de Benders, sendo este último o implementado, e do qual segue o algoritmo:

1. ENQUANTO $v_P > (1+\tau)v(P)$:

   (a) RESOLVER mestre primal;

(b) SE $v_P \leq (1+\tau)v(P)$. FIM.

(c) RESOLVER subproblema primal:

(d) GERAR corte para o mestre primal;

(e) SE $v_P < v(SP)$, atualizar incumbente $v_P \leftarrow v(SP)$

Capítulo 3

Implementação Computacional do Modelo

Neste capítulo será feita uma análise da estrutura de cada um dos problemas resultantes da aplicação da Decomposição Cruzada com o objetivo de buscar uma forma eficiente de implementação computacional.

3.1 Subproblema Primal

No caso geral o subproblema da decomposição de Benders é resolvido por um algoritmo de programação linear genérico, como o Simplex por exemplo. Muitos problemas, no entanto, têm uma estrutura especial, o que lhes permite resolução por especializações muito mais eficientes, do ponto de vista de uma resolução mais rápida que o método original.

O subproblema da decomposição de Benders pode ser transformado, através de alguma manipulação algébrica, em um Problema de Fluxo em Redes a Custo Mínimo (PFCM), cuja estrutura permite a aplicação de versão do método Simplex que considera particularmente a morfologia de uma base caracterizando-a na forma de árvore enraizada.

Algo que se procura evitar, ou pelo menos fazê-lo de um modo mais inteligente é a chamada Fase I, pois demanda grande esforço computacional a busca de uma solução básica factível, quando não se considera algum critério que permita a redução do número de arcos adicionados à rede original. Para que se tenha uma idéia do que isto representa, o número de arcos a serem adicionados pela aplicação da Fase I clássica neste subproblema primal seria $(N+1)T$, que cresce com o número de períodos e de itens ao mesmo tempo. Porém, baseando-se na estrutura da árvore e na forma da rede completa vê-se que é possível a construção de uma tal solução de partida utilizando apenas $(N+T)$ arcos, com uma redução bastante significativa.

3.1.1 O Problema de Fluxo a Custo Mínimo (PCFM)

Considere-se o subproblema primal (SP_λ) e assuma sem perda de generalidade que $I_{i,0} = 0$, $i \in I$ (Johnson e Montgomery [24]). Também será assumido que $I_{i,T} = 0, i \in I$, desde que isto é uma propriedade de uma solução ótima de (SP_λ) (prova no apêndice).

Fazendo-se $Y_{i,t} = p_i X_{i,t}$ e $H_{i,t} = p_i I_{i,t}$, pode-se escrever o problema (SP_λ) como:

$$(SP'_\lambda)\ \text{Min}_{Y,H,F,R,R',F,F'} \quad \sum_{i,t} \sigma_{i,t}\lambda_{i,t} + \sum_{i,t} a_{i,t}H_{i,t}/p_i + \sum_{i,t} v_{i,t}Y_{i,t}/p_i + \sum_t b_t R_t + \sum_t q_t F_t$$

sujeito a

$$\begin{aligned}
H_{i,t-1} + Y_{i,t} - H_{i,t} &= d_{i,t}p_i, && i \in I, t \in T && (3.1)\\
Y_{i,t} &\leq \lambda_{i,t}M_{i,t}p_i, && i \in I, t \in T && (3.2)\\
\sum_i Y_{i,t} - R_t + F_t &= -\sum_i \lambda_{i,t}s_i, && t \in T && (3.3)\\
R_t + R'_t &= l_t, && t \in T && (3.4)\\
F_t + F'_t &= m_t, && t \in T && (3.5)\\
Y_{i,t}, H_{i,t}, R_t, R'_t, F_t, F'_t &\geq 0, && i \in I, t \in T && (3.6)
\end{aligned}$$

Desde que $I_{i,0} = I_{i,T} = 0, i \in I$, tem-se de 3.1 e 3.3 que:

$$\sum_t (R_t + F_t) = \sum_{i,t} Y_{i,t} + \sum_{i,t} \lambda_{i,t}s_i - \sum_{i,t}(p_i d_{i,t} + \lambda_{i,t}s_i)$$

e de 3.4 e de 3.5:

$$\sum_t (R'_t + F'_t) = \sum_t (l_t + m_t) - \sum_{i,t}(p_i d_{i,t} + \lambda_{i,t}s_i) \qquad (3.7)$$

A Equação 3.7 é redundante e constitui o nó de equilíbrio do grafo. Esta equação e os conjuntos de restrições 3.1, 3.3, 3.4 e 3.5 correspondem a uma matriz de incidência nó-arco. As restrições 3.2 representam a capacidade, isto é, o fluxo máximo através dos *arcos* $Y_{i,t}$. A estrutura da matriz de incidência no-arco é mostrada a seguir:

$H_{i,t}$	$Y_{i,t}$	$H_{i,t+1}$	$Y_{i,t+1}$	$\cdots$	R_t	R'_t	F_t	F'_t	
1	1								$d_{i,t}p_i$
-1		1	1						$d_{i,t+1}p_i$
	-1				1		1		$\sum_{i,t}\lambda_{i,t}s_i \quad c_t$
					1	1			$= l_t$
							1	1	$= m_t$
					-1		-1		$= \sum_{i,t}(p_i d_{i,t} + \lambda_{i,t}s_i) - \sum_t (l_t + m_t) = S$

Porém, as colunas correspondentes aos $Y_{i,t}$ ainda possuem um terceiro elemento que pode ser eliminado ao considerar tais arcos como sendo capacitados entre zero e $M_{i,t}\lambda_{i,t}p_i$. A rede finalmente obtida é mostrada na Figura 3.1:

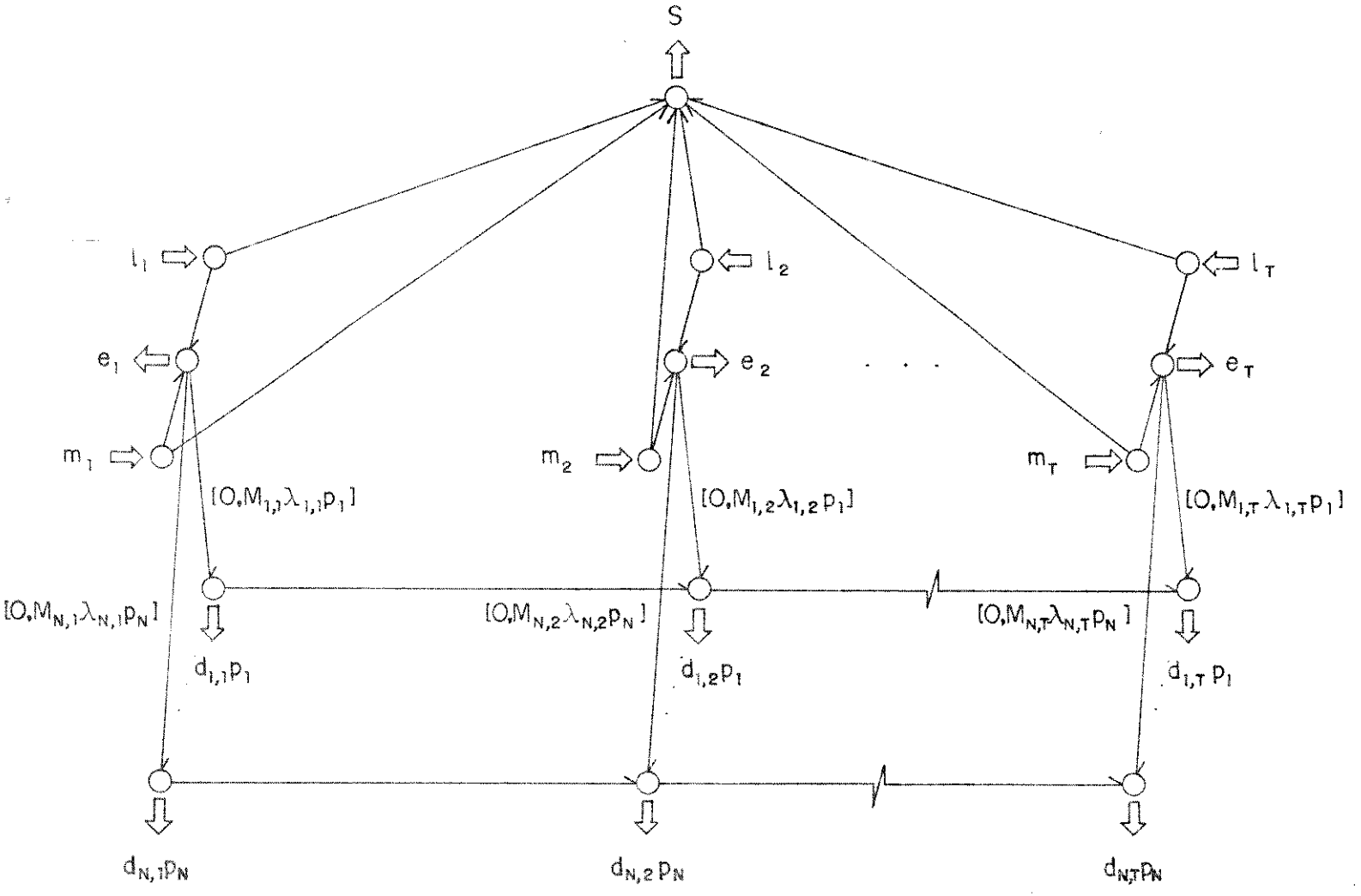

Figura 3.1.: Grafo do problema (SP_λ)

3.1.2 Busca de Uma Solução Básica Factível de Partida para o (PFCM)

Um fato muito importante que deve ser observado quando se busca uma solução básica factível de (SP_λ) para quaisquer valores de $\lambda_{i,t}$ que o problema mestre tenha proposto é que tais valores podem levar a soluções infactíveis e, como conseqüência, um raio extremo deve ser gerado ao invés de um ponto extremo. No entanto, como está sendo utilizado um algoritmo aplicado ao problema primal e não ao dual diretamente, a derivação de raios extremos constitui num esforço adicional que pode ser contornado.

Deve ser lembrado que sempre é possível limitar o politopo dual impondo limites bastante altos às suas variáveis, fazendo com que não precisem ser gerados raios extremos [20], pois estes passarão a ser considerados, indiretamente, através da combinação dos pontos extremos adicionados e dos já existentes no politopo. A regra de imposição de limites suficientemente altos a um problema, conhecida como método da Restrição Artificial M-grande[1], é equivalente à aplicação do método do M-grande às variáveis de seu dual. Portanto, basta que se aplique este último ao primal para que se garanta apenas a geração de pontos extremos. Isto será visto com mais detalhe a seguir:

Com a adição de (N+T) arcos é possível garantir uma solução factível[2] para quaisquer valores de $\lambda_{i,t}$. A construção desta solução factível é mostrada na Figura 3.2.

É fácil verificar que os arcos artificiais adicionados são suficientes para tornar o problema factível, pois os arcos que ligam ao nó de equilíbrio levam os recursos disponíveis para onde eles são necessários e os outros garantem que as demandas sejam atendidas de modo independente dos valores de $\lambda_{i,t}$. Da Equação 3.7 segue-se que se $\sum_t (l_t + m_t) \quad \sum_{i,t}(p_i d_{i,t} + \lambda_{i,t} s_i) < 0$ o problema é infactível. Portanto, uma condição necessária para a factibilidade é que o nó de equilíbrio seja um sorvedouro.

A discussão anterior pode ser formalizada da seguinte maneira:

N dos arcos artificiais correspondem às variáveis $Y'_{i,1}$, $i = 1, \ldots, I$, que são incluídas nas Equações 3.1 para $t = 1$, isto é:

$$Y_{i,1} + Y'_{i,1} \quad H_{i,1} \quad d_{i,1} p_i, i = 1, \ldots, I, \tag{3.8}$$

$$Y'_{i,1} \geq 0, i = 1, \ldots, I.$$

Os T arcos restantes correspondem às variáveis $Z_t, t = 1, \ldots, T$, incluídas nas Equações 3.3:

$$\sum_i (Y_{i,1} + Y'_{i,1}) \quad R_1 \quad F_1 \quad Z_1 \quad \sum_i \lambda_{i,1} s_i \tag{3.9}$$

$$\sum_i Y_{i,t} \quad R_t \quad F_t \quad Z_t \quad \sum_i \lambda_{i,t} s_i, t = 2, \ldots, T,$$

[1] Big-M artificial constrained method.

[2] No sentido de que qualquer infactibilidade será refletida no custo da função objetivo.

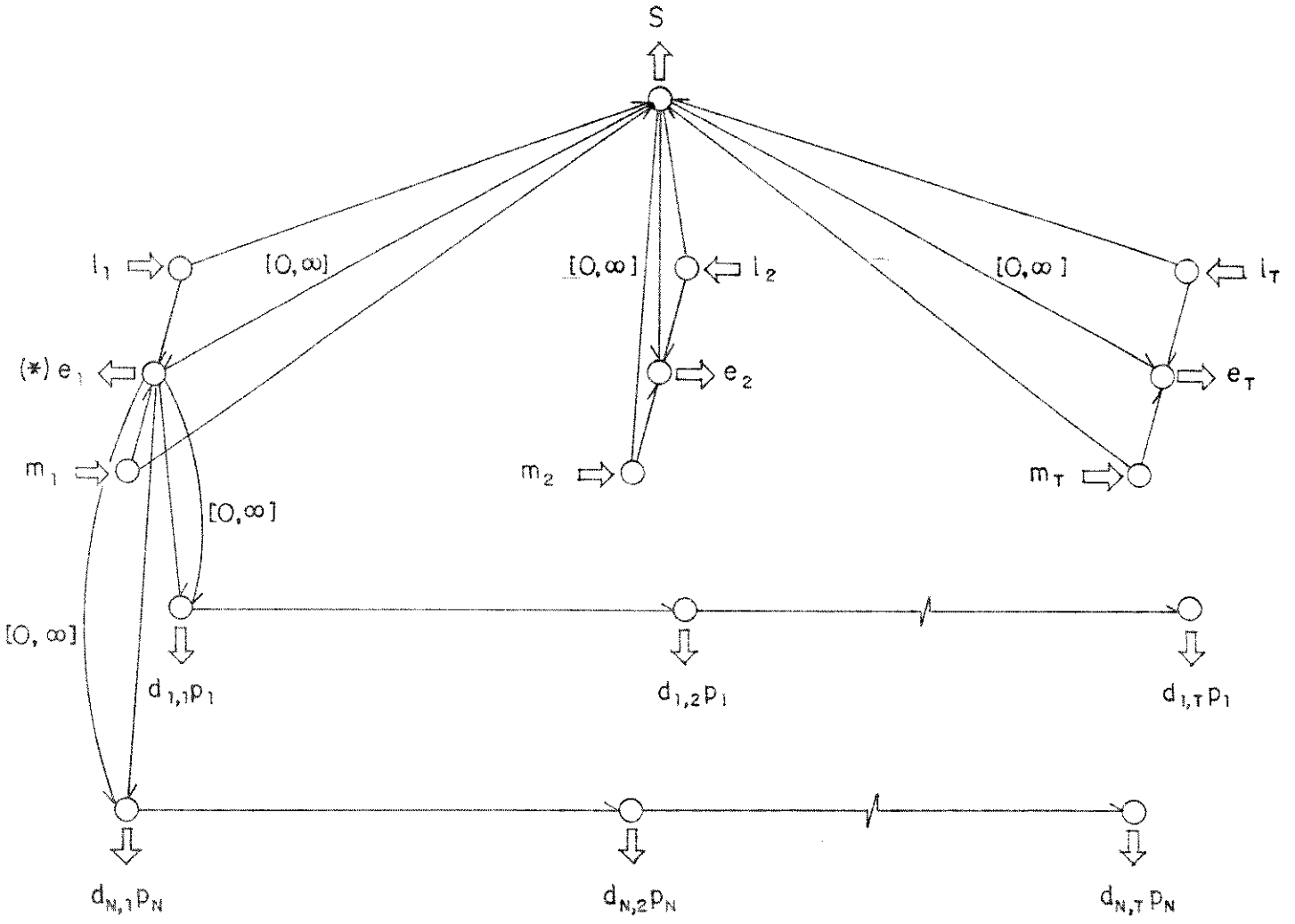

Figura 3.2: Solução básica de partida.

$$Z_t \geq 0, t \in T$$

Definindo

$$Y'_{i,1} = \sum_{t=1}^{T} d_{i,t} p_i, i \in I,$$
$$Y_{i,t} = 0, i \in I, t \in T,$$
$$R_t = F_t = 0, t \in T,$$

obtém-se a solução básica factível de partida.

A função objetivo é acrescida destes novos termos cuja utilização é penalizada com valores altos em relação aos existentes de modo a desincentivar o seu uso[3], isto é,

Minimizar $\sum_{i,t} \lambda_{i,t}\sigma_{i,t} + \sum_{i,t} a_{i,t} H_{i,t}/p_i + \sum_{i,t} v_{i,t} Y_{i,t}/p_i + \sum_i M Y'_{i,1} + \sum_t b_t R_t + \sum_t q_t F_t + \sum_t M Z_t$

onde M é um número suficientemente grande.

A influência de tais alterações nos cortes de Benders será discutida nas seções seguintes.

3.1.3 Os Cortes de Benders

Será mostrado agora que o politopo dual torna-se fechado pela aplicação do método do M-grande ao primal, eliminando dessa forma todos os raios extremos. Por sua vez, os cortes primais são construídos a partir da função objetivo dual que não sofre alteração com a aplicação do método do M-grande. Então, tais cortes permanecem inalterados. Tal idéia de inalterabilidade dos cortes não é muito intuitiva, sendo conveniente uma rápida explicação:

Quando transformado em PFCM, o dual do subproblema primal ,apresentado no Capítulo 2, assume a seguinte forma:

$$(DSP)' \ \text{Max}_{A,B,C,D,E,G} \quad \begin{array}{l} \sum_{i,t} -p_i d_{i,t} A_{i,t} + \sum_{i,t} M_{i,t} \lambda_{i,t} p_i B_{i,t} + \sum_t (\ C_t \sum_i s_i \lambda_{i,t} + l_t D_t + m_t E_t) \\ G[\sum_{i,t}(d_{i,t} p_i + \lambda_{i,t}) \quad \sum_t (m_t + l_t)] \end{array}$$

sujeito a

$$A_{i,t} - A_{i,t+1} \leq a_{i,t}/p_i, \qquad i \in I, t \in T \tag{3.10}$$
$$A_{i,T+1} = 0, \qquad i \in I \tag{3.11}$$
$$-A_{i,t} + B_{i,t} + C_t \leq v_{i,t}/p_i, \qquad i \in I, t \in T \tag{3.12}$$
$$C_t + D_t \leq b_t, \qquad t \in T \tag{3.13}$$
$$C_t + E_t \leq q_t, \qquad t \in T \tag{3.14}$$
$$D_t - G \leq 0, \qquad t \in T \tag{3.15}$$
$$E_t - G \leq 0, \qquad t \in T \tag{3.16}$$
$$B_{i,t} \leq 0, \qquad i \in I, t \in T \tag{3.17}$$

[3]caracterizando o método do M-grande.

Tornando-se o nó G como referencial na construção das bases ($G = 0$) tem-se o mesmo conjunto de restrições e a mesma função objetivo apresentadas naquele Capítulo.

Com a adição das variáveis $Y'_{i,1}$ e Z_t ao problema primal, tem-se dois novos conjuntos de restrições no dual:

$$A_{i,1} + C_1 \leq M, i \in I,$$

$$-C_t + G \leq M, t \in T.$$

Observe-se que se $G = 0$:

$$C_t \geq -M, \tag{3.18}$$

e que

$$A_{i,1} \geq C_1 - M \Rightarrow A_{i,1} \geq -2M,$$

e da Equação 3.10:

$$A_{i,t+1} \geq A_{i,t} - a_{i,t}/p_i. \tag{3.19}$$

Desde que em $(DSP)^t$, $B_{i,t} \leq 0$, $E_t \leq 0$, $D_t \leq 0$ e considerando as Equações 3.18 e 3.19 segue-se que a função objetivo de $(DSP)^t$ é limitada. Note que a região $F(DSP)^t$não é vazia, pois tomando $C_t = D_t = E_t = B_{i,t} = 0$, $A_{i,1} \geq 0$ é sempre possível factibilizar 3.10 e 3.12 fazendo-se $A_{i,t+1} = \max\{A_{i,t} - a_{i,t}/p_i, 0\}$.

Os coeficientes negativos de $A_{i,t}$ e C_t somados ao fato de que tais variáveis são irrestritas constituem uma possibilidade de crescimento ilimitado do dual que é bloqueada pelas Equações 3.18 e 3.19. Como as demais combinações coeficientes-variáveis não permitem raios extremos por estarem limitadas, conclui-se que o politopo está fechado.

A influência no corte não é feita de modo direto e sim na alteração dos valores das variáveis duais que o compõem. Assim, é suficiente a aplicação do M-grande ao primal, deixando que os valores duais indiquem ao problema mestre se a solução proposta determinava um raio extremo ou não.

3.2 Subproblema Dual

Como comentado no Capítulo 2, após a dualização das restrições que unem os blocos da matriz de restrições, o subproblema dual é separado em N problemas, um para cada produto, que podem ser resolvidos independentemente. Será mostrado nesta seção como isto acontece.

3.2.1 Formulação do Problema Lagrangeano

Tomando o dual Lagrangeano do problema (P) em relação às Restrições 2.3, tem-se[4]:

$$(SD_C)\ \text{Min}_{Y,H,R,F,\lambda}\ \sum_{i,t}(\sigma_{i,t} - s_i C_t)\lambda_{i,t} + \sum_{i,t} a_{i,t} H_{i,t}/p_i + \sum_{i,t}(v_{i,t}/p_i - C_t)Y_{i,t} + \sum_t (b_t + C_t)R_t + \sum_t (q_t + C_t)F_t$$

sujeito a

$$H_{i,t-1} + Y_{i,t} - H_{i,t} = d_{i,t}p_i, \qquad i \in I, t \in T \tag{3.20}$$

$$Y_{i,t} \leq \lambda_{i,t} M_{i,t} p_i, \qquad i \in I, t \in T \tag{3.21}$$

$$R_t \leq l_t, \qquad t \in T \tag{3.22}$$

$$F_t \leq m_t, \qquad t \in T \tag{3.23}$$

$$Y_{i,t}, H_{i,t}, R_t, F_t \geq 0, \qquad i \in I, t \in T \tag{3.24}$$

$$\lambda_{i,t} \in \{0,1\} \qquad i \in I, t \in T \tag{3.25}$$

As variáveis R_t e F_t não estão ligadas às equações de balanço do problema, portanto podem ser determinadas, de modo independente, através de seus custos:

$$R_t \begin{cases} l_t & \text{se } (b_t + C_t) < 0 \\ 0 & \text{caso contrário} \end{cases} \tag{3.26}$$

$$F_t \begin{cases} m_t & \text{se } (q_t + C_t) < 0 \\ 0 & \text{caso contrário} \end{cases} \tag{3.27}$$

Após a definição dos valores de R_t e F_t, tem-se:

$$v(SD_C) = \sum_t [(b_t + C_t)R_t + (q_t + C_t)F_t] + v(SD_C)'$$

onde:

$$(SD_C)'\ \text{Min}_{Y,H,\lambda}\ \sum_{i,t}(\sigma_{i,t} - s_i C_t)\lambda_{i,t} + \sum_{i,t} a_{i,t} H_{i,t}/p_i + \sum_{i,t}(v_{i,t}/p_i - C_t)Y_{i,t}$$

sujeito a

$$H_{i,t-1} + Y_{i,t} - H_{i,t} = d_{i,t}p_i, \qquad i \in I, t \in T \tag{3.28}$$

$$Y_{i,t} \leq \lambda_{i,t} M_{i,t} p_i, \qquad i \in I, t \in T \tag{3.29}$$

$$Y_{i,t}, H_{i,t} \geq 0, \qquad i \in I, t \in T \tag{3.30}$$

$$\lambda_{i,t} \in \{0,1\} \qquad i \in I, t \in T \tag{3.31}$$

[4]Novamente os estoques inicial e final são considerados nulos.

Analisando o conjunto de restrições e a função objetivo, percebe-se que o problema não possui ligação em i e, portanto, pode ser resolvido separadamente para cada produto:

$$(SD_C)_i' \ \text{Min}_{Y,H,\lambda} \quad \sum_t (\sigma_{i,t} \quad s_i C_t)\lambda_{i,t} + \sum_t a_{i,t} H_{i,t}/p_i + \sum_t (v_{i,t}/p_i \quad C_t) Y_{i,t}$$
$$\text{sujeito a}$$

$$\begin{aligned}
H_{i,t\ 1} + Y_{i,t} - H_{i,t} &= d_{i,t} p_i, & i \in I, t \in T \quad (3.32)\\
Y_{i,t} &\leq \lambda_{i,t} M_{i,t} p_i, & i \in I, t \in T \quad (3.33)\\
Y_{i,t}, H_{i,t} &\geq 0, & i \in I, t \in T \quad (3.34)\\
\lambda_{i,t} &\in \{0,1\} & i \in I, t \in T \quad (3.35)
\end{aligned}$$

Evidentemente, uma solução factível para $(SD_C)_i'$ existirá se, e somente se,

$$\sum_{j=1}^{t} M_{i,j} \geq \sum_{j=1}^{t} d_{i,j}, t \in T$$

desde que $H_{i,0} = H_{i,T} = 0$.

A imposição da ocorrência de um custo de preparação para a produção de um dado item, expressa na Equação 3.33, pode ser relaxada para

$$Y_{i,t} \leq M_{i,t} p_i \quad (3.36)$$

se a componente da função objetivo em $Y_{i,t}$ considerar que a preparação ocorreu todas as vezes em que $Y_{i,t} > 0$. Isto pode ser feito transformando-se a componente em $Y_{i,t}$ da função objetivo para

$$F_t(Y_{i,t}) \quad \begin{cases} (\sigma_{i,t} \quad s_i C_t) + (v_{i,t}/p_i \quad C_t) Y_{i,t} & \text{se } Y_{i,t} > 0 \\ 0 & \text{caso contrário} \end{cases} \quad (3.37)$$

Como mostrado no Apêndice, não existe perda de generalidade em restringir C_t a valores não positivos e, portanto, a função representada na Figura 3.3 é côncava.

Portanto, o problema $(SD_C)_i'$ pode ser reformulado como:

$$\begin{array}{cc} (SD_C)_i'' \ \text{Min}_{Y,H} & \sum_t [F_t(Y_{i,t}) + a_{i,t} H_{i,t}/p_i] \\ \text{sujeito a} & (3.32), (3.36) \text{ e } (3.34) \end{array}$$

Também se observa que o conjunto de restrições determina uma rede capacitada em $Y_{i,t}$ e com função de custo côncava, desde que lhe seja incluído um nó de equilíbrio com uma fonte $\sum_t d_{i,t} p_i$. Esta rede é mostrada na Figura 3.4.

Alguns métodos de solução para $(SD_C)_i'$ são discutidos na Seção seguinte.

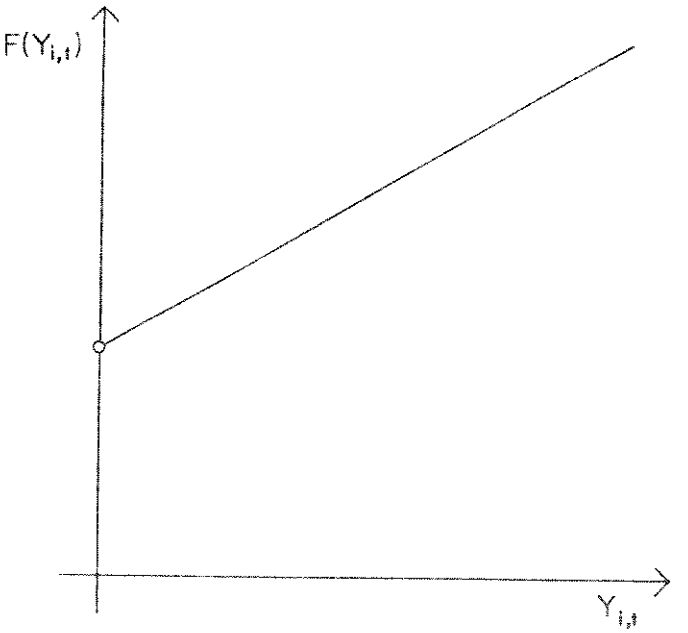

Figura 3.3: Função de custo para $Y_{i,t}$.

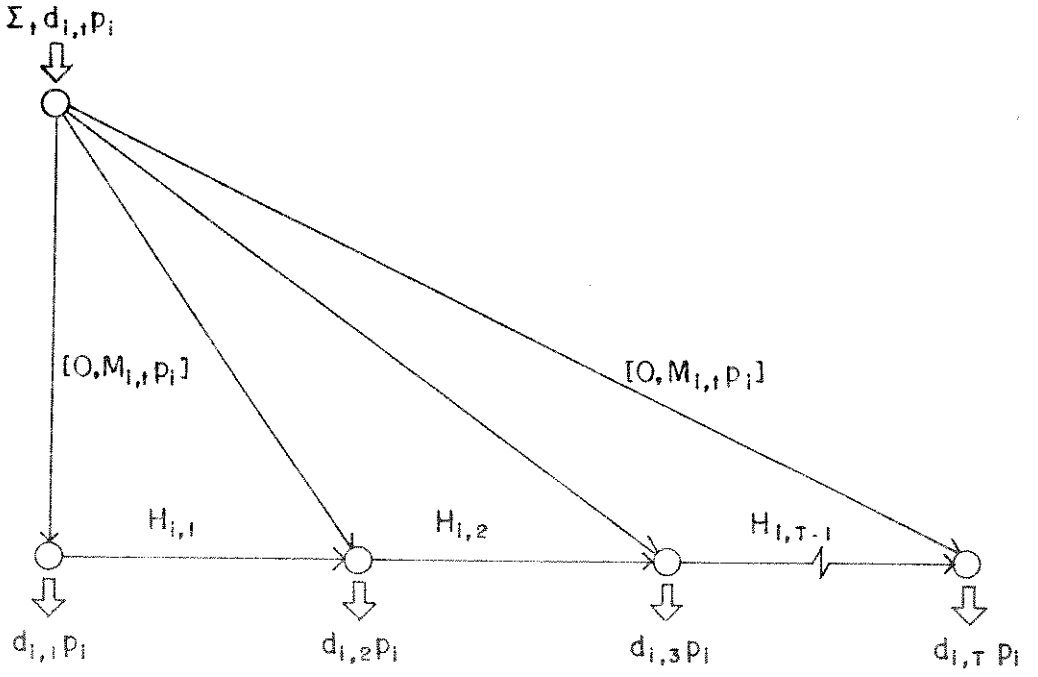

Figura 3.4. Grafo completo para o subproblema dual.

3.2.2 Resolução de Subproblema Dual

Sendo um problema de Programação Inteira Mista Capacitado, a complexidade envolvida na resolução do subproblema dual é quase tão elevada quanto à do problema completo. Como demonstrado em [2] o problema é *NP-hard* e algoritmos de resolução em tempo polinomial só serão possíveis quando:

(a) As capacidades são todas iguais, reduzindo a complexidade para $O(n^4)$;

(b) As capacidades são suficientemente elevadas, podendo o problema ser resolvido pelo algoritmo de Wagner-Within [1], de complexidade $O(n^2)$.

Para a resolução deste problema de custos côncavos, são encontradas na literatura algumas formas de abordagem explorando propriedades analíticas desta classe de problemas. No entanto, a eficiência desses algoritmos depende do conjunto de dados do problema, conforme é discutido a seguir:

Uma abordagem por *Desdobramento e Sondagem* foi proposta em [19]. Esta proposta é crítica em relação ao custo de preparação, pois quando este custo é elevado, os limites inferior e superior são enfraquecidos, isto é, não dão estimativas suficientemente boas para a sondagem dos nós de modo rápido. O enfraquecimento dos limites podem ser vistos facilmente analizando-se o gráfico da Figura 3.5:

Supondo-se que $\overline{Y}_{i,t}$ seja o valor ótimo para o problema, com a integralidade de $\lambda_{i,t}$ relaxada, o ponto 1 fornece um *limitante inferior* da função de custo e o 2, um *limitante superior*. Então, quanto mais próximos um do outro, mais fácil será a sondagem daquele nó. Porém, quanto maior o custo de preparação, mais distantes os limites estarão entre si e mais difícil será a sondagem.

Uma forma de resolução através de programação dinâmica é desenvolvida em [23], onde as propriedades da solução básica são exploradas no desenvolvimento de um algoritmo de busca em árvores e certas condições utilizadas na redução do esforço desta busca.

Abordagem semelhante em programação dinâmica é encontrada em [2], onde a estrutura dos planos ótimos de produção é estudada com o objetivo de melhorar o algoritmo padrão (de programação dinâmica), fazendo com que este seja reduzido a um algoritmo de Programação Dinâmica em Caminho Mínimo. Neste artigo, os testes indicam uma redução do tempo computacional da ordem de 10, para uma demanda média de 20 com até 12 períodos e uma redução pela metade em um horizonte de 24 períodos. Com o crescimento da demanda média para 200, a vantagem de sua utilização é ainda maior. Tais testes também mostram que esta abordagem e a encontrada em [23] requerem aproximadamente o mesmo tempo em testes com até 24 períodos.

A conclusão a que se chega após o estudo das várias propostas para a resolução do problema é que a escolha deve estar dirigida para a classe de problemas a serem resolvidos,

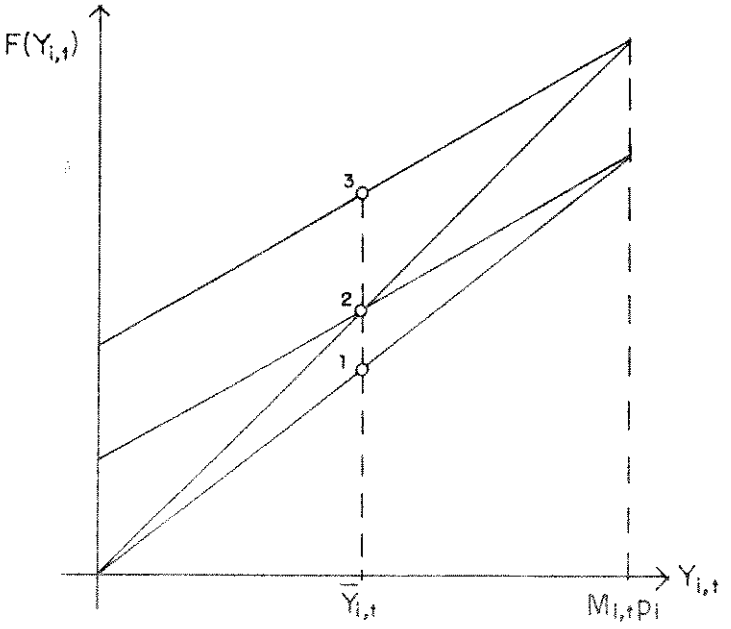

Figura 3.5.: Aproximação da função côncava utilizada no desdobramento e sondagem.

ou seja, dependendo do conjunto de dados um algoritmo pode se comportar melhor que outro. Um exemplo disto é o custo de preparação que, quando elevado, dificulta a sondagem com o método de Desdobramento e Sondagem. Baseando-se na existência de problemas com custos de preparação significativos, justifica-se parcialmente a utilização do último algoritmo comentado.

3.2.3 O Algoritmo de Programação Dinâmica Modificado

A forma utilizada no densevolvimento do algoritmo de programação dinâmica modificado segue a mesma filosofia daquela proposta por Florian e Klein [2].

Na introdução de uma notação necessária ao densevolvimento do algoritmo é conveniente que seja suprimido o índice i, correspondente aos produtos, para evitar o acúmulo desnecessário de índices.

Considere-se, então, para cada $t \in T$ as seguintes quantidades acumuladas:

$$Q_t \qquad \sum_{j=1}^{t} d_j p \tag{3.38}$$

$$U_t \qquad \sum_{j=1}^{t} M_j p \tag{3.39}$$

$$V_t \qquad \sum_{j=1}^{t} Y_j \tag{3.40}$$

A função de custo conforme definida anteriormente é simplificada para $F_t(Y_t)$ e a função de estocagem para

$$h_t(H_t) = a_t H_t / p \tag{3.41}$$

Desta forma, a função de custo total é dada por:

$$\sum_{t} [F_t(Y_t) + h_t(H_t)]$$

Sejam $D_t(V)$ o custo de um plano de produção ótimo sobre os períodos $1, \ldots, t$ sujeito a $V_t = V$, Γ_t o conjunto de níveis factíveis de produção cumulativa no período t e $\chi_t(V)$ o conjunto de quantidades de produção factíveis Y_t no período t sujeito a $V_t = V$.

O custo do plano ótimo de produção, igual a $D_T(Q_T)$ é calculado de acordo com a seguinte recursão:

$$D_0(V) \quad \begin{cases} 0 & \text{se } V = 0 \\ \infty & \text{caso contrário} \end{cases} \tag{3.42}$$

$$D_t(V) = \begin{cases} min_{Y \in X_t(V)}\{D_{t-1}(V \quad Y) + F_t(Y)\} + h_t(V \quad Q_t) & \text{se } V < \Gamma_t \\ \infty & \text{caso contrário} \end{cases} \tag{3.43}$$

Os valores correspondentes de Y_t são obtidos através de *backtracking* após determinação de todos os planos parciais $D_t(V \quad Y), t < T$ que levaram à solução ótima. Este é um algoritmo geral de programação dinâmica, portanto não considera nenhuma particularidade dos planos ótimos de produção. Tais particularidades serão exploradas doravante na busca de uma melhoria no algoritmo.

A propriedade de Decomposição do Estoque, que será vista a seguir, determina o ponto de partida na construção deste novo algoritmo:

Teorema 3.1 *Suponha que a restrição*

$$H_k = 0, \quad \textit{para algum} \quad K < i, ..., T \quad 1,$$

é adicionada ao problema e se

$$\sum_{j=k+1}^{t} M_j \geq \sum_{j=k+1}^{t} d_j, t \quad k+1, ..., T \quad 1,$$

então, uma solução ótima para o problema original pode ser encontrada pela busca de soluções independentes para os primeiros k períodos e para os últimos $T \quad k$ ***períodos.***

Prova: Ver [18].

Uma conseqüência imediata do Teorema 3.1 é a possibilidade da construção de um algoritmo de programação dinâmica recursivo, utilizando os períodos $0, \ldots, T$ como estados:

$$\begin{aligned} D_0^* &= 0 \\ D_m^* &= \min_{0 \leq l < m}\{D_l^* + E_{l,m}\}, m = 1, \ldots, T \end{aligned} \tag{3.44}$$

onde:

$E_{l,m}$ é o custo associado com a aplicação de um plano ótimo sobre os períodos $l+1, \ldots, m$, onde $H_l = H_m = 0$ e $H_t > 0$ para todo $t = l+1, \ldots, m \quad 1$,

D_m^* é o custo associado ao plano ótimo de produção sobre os períodos $1, \ldots, m$.

O conjunto de restrições de $(SD_t)''$ definem um conjunto convexo limitado. O problema, então consiste da minimização de uma função côncava sobre um politopo convexo limitado, e como tal, tem seu mínimo atingido em um ponto extremo.

Note-se, no entanto, que sem a descrição dos pontos extremos, este algoritmo seria inútil, pois o cálculo de cada um dos $(T+1)T/2$ valores $E_{l,m}$ poderia ser um problema quase tão difícil quanto o original. A caracterização de pontos extremos será feita em 3.2.4, a seguir.

3.2.4 Caracterização dos Pontos Extremos

Seja

$$S_{u,v} \quad \{Y_t, t \quad u+1,\ldots,v \mid H_u \quad H_v \quad 0; H_t > 0 \text{ para } u < t < v\}, 0 \le u < v \le T$$

uma seqüência de produção.

Definição 3.1 *Uma suqüência de produção $S_{u,v}$ é capacitada se o nível de produção em no máximo um período t, $(u+1 \le t \le v)$ é positivo e menor que a capacidade, isto é, $0 < Y_t < M_t p$, e todos os outros níveis estão em zero ou na capacidade.*

No Teorema 3.2, a seguir, é mostrado que a definição das seqüências capacitadas nada mais é que a caracterização de uma solução básica em um grafo dirigido.

Teorema 3.2 *Um plano factível $(Y_1,\ldots,Y_T)$ está no conjunto de pontos extremos se, e somente se, consiste de seqüências capacitadas.*

Prova:

Basta que seja mostrado que um conjunto de seqüências capacitadas é equivalente a um ponto extremo do politopo. Como já foi visto, o conjunto de restrições representam um grafo dirigido. Como um ponto extremo em um grafo dirigido é uma árvore enraizada [25], este não pode conter ciclos. Veja-se, por exemplo, na Figura 3.6 o grafo dividido em duas seqüências. Como os arcos correspondentes aos estoques H_1, H_3 e H_4 têm fluxo estritamente positivos (pela própria definição de seqüência de produção) e não são capacitados, estão obrigatoriamente na base. Portanto, para que não haja ciclo apenas um dos arcos correspondentes a Y_t deve estar na base, isto é, $0 < Y_t < M_t p$. Como as bases podem ser degeneradas, a definição impõe que no máximo um *deles esteja estritamente entre os limites $0 < Y_t < M_t p$, completando a prova.*

Do Teorema 3.2 segue-se que $E_{l,m}$ é o valor da solução ótima do problema $P_{l,m}$, com $0 < l < m < T$ que pode ser reformulado como:

$$(P_{l,m}) \ \text{Min}_{\lambda,X,H} \quad \sum_t |F_t(Y_t) + h_t(H_t)|$$
sujeito a

$$H_{t-1} + Y_t - H_t = d_t p, \qquad t < T \tag{3.45}$$

$$0 < Y_f < M_f p, \quad \text{para no máximo um } f,\ l+1 \le f \le m \tag{3.46}$$

$$0 < Y_t < M_t p, \quad \text{para } l+1 < t < m \tag{3.47}$$

$$Y_t \in \{0, M_t p\} \quad \text{para } t \neq f \tag{3.48}$$

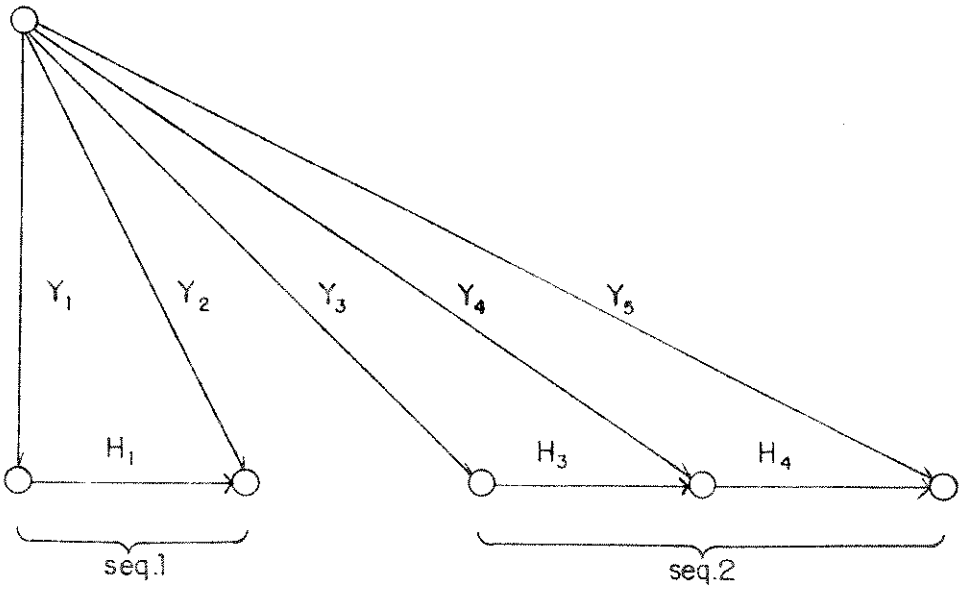

Figura 3.6.: Grafo correspondente às seqüências capacitadas.

Como em uma seqüência $S_{l,m}$ só existem $m - l$ possibilidades de arco fracional, pode-se resolver $P_{l,m}$ fixando-se o período fracional f, obtendo-se o problema $P^f_{l,m}$, cujo valor ótimo é denotado por $E^f_{l,m}$. Tendo-se, então os valores $E^f_{l,m}$, calcula-se $E_{l,m}$ através de:

$$E_{l,m} = \min_{l+1 \leq f \leq m}\{E^f_{l,m}\} \tag{3.49}$$

A aplicação da recursão 3.44 corresponde à busca de um caminho mínimo no grafo[5] da Figura 3.7.

Seja $S^f_{l,m}$ a seqüência capacitada em que apenas o nível de produção f pode estar entre 0 e $M_f p$. Daí, o custo $E^f_{l,m}$ é o custo do plano de produção ótimo sobre a seqüência $S^f_{l,m}$.

Uma forma trivial de encontrar $E^f_{l,m}$ seria enumerar todas as possibilidades para os níveis de produção nos períodos diferentes de f, verificar se é possível factibilizá-los com os Y_f e determinar o de menor custo. Isto, porém, demandaria um esforço desnecessário pois é possível estabelecer algumas condições de factibilidade *a priori*:

Dado que a seqüência $S^f_{l,m}$ tem, por definição, estoques inicial e final nulos, os níveis acumulados de produção factíveis em l e m são Q_l e Q_m respectivamente, isto é,

$$\Gamma_l = \{Q_l\}$$

$$\Gamma_m = \{Q_m\}$$

Para qualquer $l < t < f$ o nível V_t será factível somente se houver $V_{t-1} \in \Gamma_{t-1}$ tal que

$$V_t = V_{t-1} + M_t p$$

e/ou

$$V_t = V_{t-1} + 0$$

sejam factíveis. Para que isto ocorra é necessário que o nível de produção V_t:

(a) Seja maior que a demanda naquele período, pois não pode haver estoque nulo entre $t-1$ e t:

$$V_t > Q_t$$

(b) Adicionado às capacidades em $t+1, \ldots, m$ seja suficiente para satisfazer à demanda cumulativa Q_m:

$$V_t + Q_m \geq (U_m - U_t)$$

(c) Não ultrapasse o nível acumulado requerido no final da seqüência $H_m = 0$.

$$V_t \leq Q_m$$

[5] Representado para 4 períodos

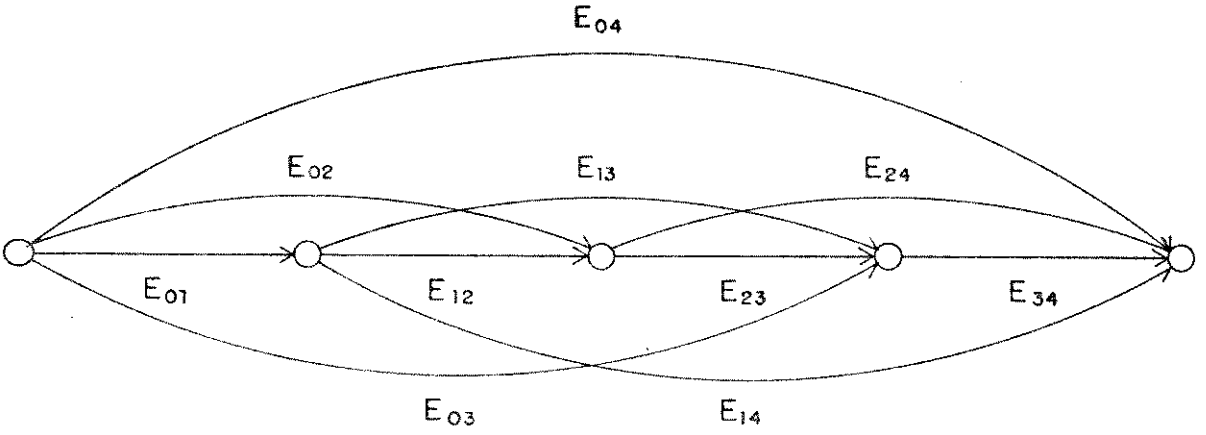

Figura 3.7.: Grafo correspondente ao algoritmo de programação dinâmica.

Este é um processo *para frente* onde se prevê a factibilidade de uma proposta V_t dependendo de V_{t-1}. Caso esta proposta seja aceita, ela fará parte do conjunto de níveis factíveis de produção acumulada do período t, formando uma árvore binária, pois de cada $V_{t-1} \in \Gamma_{t-1}$, dois outros níveis V_t podem ser factíveis. Esta árvore é ilustrada na Figura 3.8(a).

Para se fazer uma previsão sobre os níveis V_t, $f \leq t < m$ a partir do nível a ser atingido no final, realiza-se uma análise *para trás* onde:

$$V_t = V_{t+1} - M_{t+1}p$$

e/ou

$$V_t = V_{t+1}$$

são factíveis se safisfazem às seguintes restrições:

(a) V_t é maior que a demanda em t pois não pode haver estoque nulo entre t e $t+1$:

$$V_t \geq Q_t$$

(b) V_t não deve exceder a capacidade atual de produção acumulada:

$$V_t \leq U_t$$

Desde que para cada $V_{t+1} \in \Gamma_{t+1}$ dois V_t podem ser factíveis temos uma outra árvore binária, porém, desta vez, *para trás* conforme mostrado na Figura 3.8(b).

Define-se como um *ponto terminal* aquele que, na árvore *para frente* conseguiu atingir o nível $t = f - 1$, ou na árvore *para trás* atingiu o nível f. Um exemplo de uma árvore com seus pontos terminais é mostrado na Figura 3.9. Os arcos que estão faltando nesta figura representam arcos infactíveis.

Agora, seja um par de pontos terminais uma combinação de um ponto da árvore *para frente* e um da árvore *para trás*. Então, as seqüências de produção factíveis são determinadas daqueles pares de pontos terminais para os quais:

$$0 \leq V_f - V_{f-1} \leq M_f p$$

Após a identificação dos pares de pontos terminais factíveis, são calculados os custos das rotas que conectam tais pontos e a de menor custo é escolhida. Desta rota, a seqüência ótima de produção é determinada através de:

$$Y_t = V_t - V_{t-1} \quad t = 1, \ldots, m$$

Note, porém, que esta não é necessáriamente a melhor seqüência $S_{1,m}$. Esta somente será conhecida após aplicada a Equação 3.49. Encontrados todos os $E_{1,m}$, o caminho mínimo é determinado pelo algoritmo de programação dinâmica 3.44.

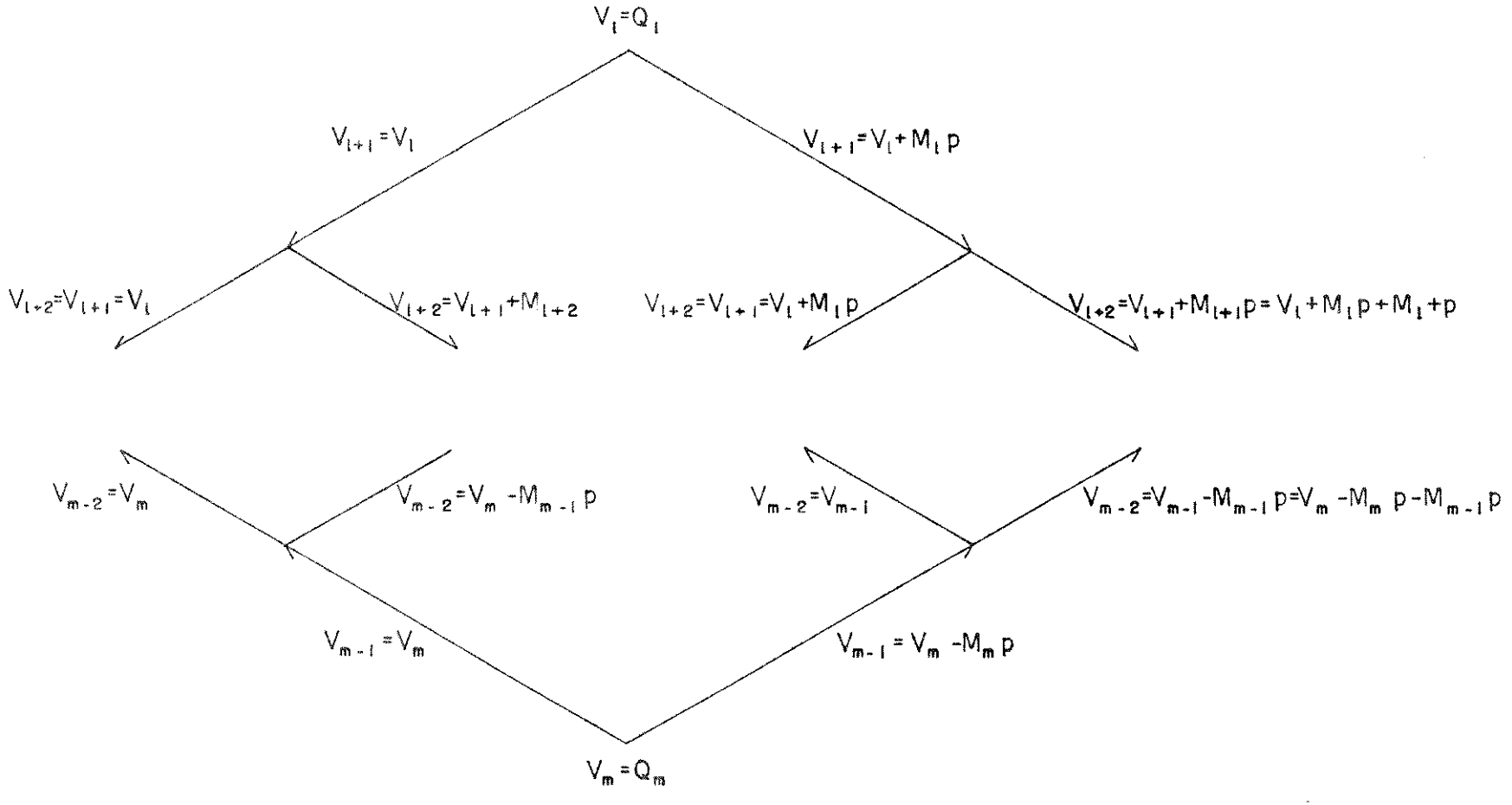

Figura 3.8. a.: Árvore binária para a frente.

b.: Árvore binária para trás.

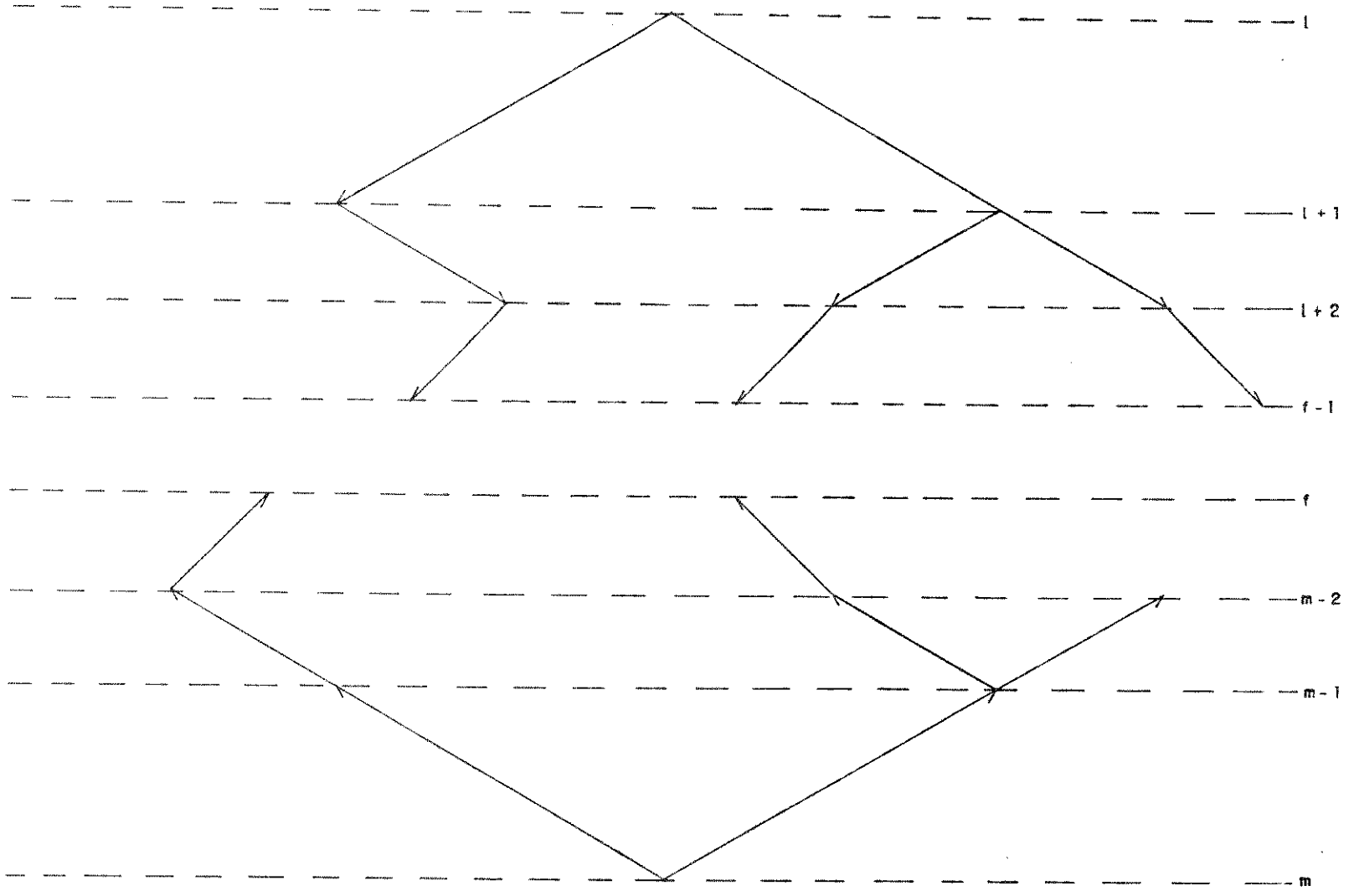

Figura 3.9.: Exemplo de uma árvore com os seus pontos terminais.

3.3 Problema Mestre Primal

O problema mestre da decomposição de Benders, que pode ser escrito na forma:

$$(PB)\ \text{Min}_{z,\lambda} \quad z$$
sujeito a

$$z \geq b_i + A_i\lambda \qquad i = 1,\ldots,n \tag{3.50}$$
$$0 \geq b_i + A_i\lambda \qquad i = n+1,\ldots,n+m \tag{3.51}$$
$$z \quad \text{irrestrito} \tag{3.52}$$

é um problema de programação inteira mista com apenas um variável real (z), onde b é um vetor $(m+n)$, A é uma matriz $(m+n)\times p$ e λ um vetor de ordem p.

Algoritmos e estratégias de resolução para este problema são raros: uma estratégia proposta em [13] sugere que o problema mestre não precisa ser resolvido até a otimalidade. Ao invés disto, procura-se uma solução cujo valor da função objetivo é menor que o incumbente primal. Um procedimento heurístico é utilizado para procurar uma boa solução factível de partida e, possivelmente, resultar em uma convergência mais rápida.

Vários métodos heurísticos são propostos em [26] objetivando aliviar a carga computacional imposta pelas sucessivas resoluções do problema mestre, que também não é resolvido até a otimalidade.

No contexto da Decomposição Cruzada as resoluções dos problemas mestres são feitas, como já se sabe, somente quando cessa o progresso no ping-pong. Experiências computacionais mostraram que uma solução sub-ótima do problema mestre pode ser, na maioria dos casos, desinteressante, pois tal resolução não consegue propor uma boa solução nas variáveis inteiras ao subproblema primal, de modo a restabelecer o ping-pong.

Para eliminar tais problemas, decidiu-se pela elaboração de um algoritmo que explorasse as particularidades da estrutura de (PB) e que também levasse em conta o tipo de variáveis inteiras do problema, que são binárias. O ponto de partida foi o algoritmo de Balas [27] com o procedimento de *backtracking* sugerido por Geoffrion [14]. Como será visto adiante, este algoritmo permite a adoção de várias estratégias distintas de finalização, o que o torna bastante versátil e de possível utilização em outros problemas cujas variáveis inteiras são binárias.

Na elaboração do algoritmo algumas definições são necessárias:

Seja uma atribuição de valores binários a um subconjunto de λ uma *Solução Parcial* S:

$$S = \{s_1, s_2, \ldots, s_k\}, k = p,$$

onde

$$s_j \quad \left\{ \begin{array}{ccc} +j & \text{se} & \lambda_j = 1 \\ -j & \text{se} & \lambda_j = 0 \end{array} \right\}, j = 1, \ldots, k.$$

As variáveis λ_j às quais nenhum valor é atribuido não pertencem a S, sendo chamadas *variáveis livres*. Quando uma ou mais destas variáveis têm seus valores fixados, a solução parcial resultante, construida a partir de S, é definida como *solução derivada* de S.

Seja z um imcumbente para o problema, isto é, o melhor valor para (PB) obtido até um dado instante.

O algoritmo busca, dado o imcumbente $\bar{z}$ e uma solução parcial S, uma solução factível que satisfaça 3.50 e 3.51 para um valor $z < \bar{z}$, isto é, procura-se satisfazer:

$$z > z \geq b_i + A_i\lambda, \qquad i = 1, \ldots, n, \tag{3.53}$$

$$0 \geq b_i + A_i\lambda, \qquad i = n+1, \ldots, n+m. \tag{3.54}$$

Para que se tenha uma idéia da distância da atual solução parcial S à factibilidade define-se:

$$y_i^S \equiv \sum_{j \in S} A_{i,j}\lambda_j + b_i, \qquad i = n+1, \ldots, n+m, \tag{3.55}$$

$$y_i^S \equiv \sum_{j \in S} A_{i,j}\lambda_j + b_i \quad z, \; i = 1, \ldots, n. \tag{3.56}$$

se $y_i^S \leq 0$ para algum $i = n+1, \ldots, n+m$ ou $y_i^S < 0$ para algum $i = 1, \ldots, n$, a restrição i é factível na completação com zeros da solução parcial, como se pode observar nas Restrições 3.53 e 3.54. Calculadas as distâncias à factibilidade de cada restrição, forma-se um conjunto composto pelos índices das que ainda estão infactíveis:

$$I^s \equiv \{i \mid y_i^S \geq 0, i = 1, \ldots, n, y_i^S > 0, i = n+1, \ldots, n+m\} \tag{3.57}$$

Se $I^s = \emptyset$, tem-se que uma solução factível com $z < \bar{z}$ foi encontrada na completação com zeros[6]. No entanto, esta solução parcial não pode ser ainda considerada como sondada, pois ainda pode haver uma solução derivada com valor ainda menor. Deve-se, então, atualizar o incumbente[7], fazendo-se

$$\bar{z} = \max\{b_i + \sum_{j \in S} A_{i,j}\lambda_j\}, i = 1, \ldots, n \tag{3.58}$$

e verificar se existe uma solução derivada de S com solução melhor que a obtida.

Caso $I^s \neq \emptyset$ deve-se procurar reduzir a infactibidade, ou seja, em qualquer restrição $i \in I^s$ deve-se procurar um ou mais $A_{i,j} < 0$ entre as variáveis livres ($j \notin S$).

Seja então T^S o conjunto de variáveis livres que podem factibilizar alguma restrição $i \in I^s$:

$$T^s \equiv \{j \text{ livres e } A_{i,j} < 0 \text{ para algum } i \mid i \in I^s\} \tag{3.59}$$

A solução parcial S será sondada se $T^s \neq \emptyset$, pois não existe nenhuma possibilidade desta ser factibilizada. Sintetizando, temos as seguintes condições:

[6] Todas as variáveis livres com valor 0.

[7] Será explicado com mais detalhes a seguir.

(a) $I^S = \emptyset$: o incumbente deve ser atualizado;

(b) $I^S \neq \emptyset \wedge T^S = \emptyset$: solução parcial não pode ser factibilizada e está, portanto, sondada;

(c) $I^S \neq \emptyset \wedge T^S \neq \emptyset$: deve-se prosseguir ampliando a solução parcial.

Contudo, é possível estabelecer critérios mais fortes sem nenhum acréscimo de memória e com pequeno esforço computacional, observando-se que:

(c.1) Os valores negativos livres são insuficientes para factibilizar uma dada restrição, como pode ser visto a seguir:

Supondo $S = \{1, -2\}$, a restrição

$$0 \geq 25 - 10\lambda_1 - 35\lambda_2 - 10\lambda_3$$

possui $T^S = \{3\} \neq \emptyset$ no entanto é infactível, pois mesmo com $\lambda_3 = 1$ a restrição continua infactível. No caso geral basta que se some, para cada $i \in I^S$, todos os coeficientes $A_{i,j} < 0$ a y_i^S. Se o resultado ainda é infactível aquela solução parcial também o é. Então, se

$$y_i^S + \sum_{j \in T^S} \min(0, A_{i,j}) > 0, \text{para algum } i = 1, \ldots, n \mid y_i^S > 0 \qquad (3.60)$$

ou

$$y_i^S + \sum_{j \in T^S} \min(0, A_{i,j}) > 0, \text{para algum } i = n+1, \ldots, n+m \mid y_i^S > 0 \qquad (3.61)$$

a solução parcial S é infactível e está *sondada*.

(c.2) Ainda utilizando os cálculos feitos acima, é possível estabelecer um critério mais forte, pois adicionando a variável λ_4 à restrição acima com coeficiente -5:

$$0 \geq 25 - 10\lambda_1 - 35\lambda_2 - 10\lambda_3 - 5\lambda_4$$

observa-se que esta restrição é factível para $S = \{1, -2\}$, pois

$$T^S = \{3, 4\}$$

e

$$15 + \min(0, -10) + \min(0, -5) = 0$$

Porém, já se sabe de antemão que os valores de λ_3 e λ_4 têm que ter valor 1 para garantir a factibilidade.

Se além de λ_4 houvesse λ_5 com coeficiente positivo, também se saberia antecipadamente que $\lambda_5 = 0$ pois não existe nenhum outro coeficiente negativo capaz de factibilizar $\lambda_5 = 1$. Portanto, as diferenças entre os dois lados das Equações 3.60 e 3.61 dão uma medida de quão críticas estas estão em relação à factibilidade e, a partir delas, pode-se montar um outro teste, o *teste de ampliação*:

Se, para $k \notin S$

$$y_i^S + \sum_{j \notin S} \min(0, A_{i,j}) + \mid A_{i,k} \mid \geq 0, \text{para algum } i = 1, \ldots, n \tag{3.62}$$

ou

$$y_i^S + \sum_{j \notin S} \min(0, A_{i,j}) + \mid A_{i,k} \mid > 0, \text{para algum } i = n+1, \ldots, n+m \tag{3.63}$$

então:

$$\lambda_k = \begin{cases} 1 & \text{se } A_{i,k} < 0 \\ 0 & \text{se } A_{i,k} > 0 \end{cases}$$

Note-se que nas Desiqualdades 3.62 $y_i^S + \sum_{j \notin S} \min(0, A_{i,j}) < 0$, e nas 3.63 $y_i^S + \sum_{j \notin S} \min(0, A_{i,j}) < 0$, pois, caso contrário, as Desigualdades 3.60 e 3.61 se verificariam.

Após uma série de ampliações, deve-se verificar em (c.1) se o problema continua infactível e se ainda pode ser factibilizado, pois como o teste de ampliação vê apenas uma restrição de cada vez, uma ampliação numa delas pode causar infactibilidade em outra. Portanto, deve-se ficar iterando em (c.1) e (c.2) até que não haja mais ampliações.

Quando não houver ampliação e a solução parcial ainda não for completa deve-se desdobrar em $j \in T^s$ para que a solução seja ampliada. Os critérios de escolha do elemento de T^s podem ser:

(a) O primeiro elemento de T^S;

(b) O elemento $j_0 \in T^S$ correspondente ao $\min_j |\min(0, A_{ij})|$ para a restrição i mais infactível;

(c) O elemento $j_0 \in T^S$ que reduza a infactibilidade total do conjunto de restrições apresentado em [14]:

$$\min_{j \in T^S} \left| \sum_i \max(y_i^S + A_{i,j}, 0) \right| \tag{3.64}$$

Não há garantia de que a solução derivada de qualquer dos itens acima seja factível, portanto a escolha deve ser feita analisando-se o comportamento de cada uma com os dados dos problemas. Quando da programação do algoritmo, o critério (a) foi utilizado por ser o que representa o menor esforço computacional a curto prazo. Posteriormente, serão feitos testes com outros critérios analizando-os de forma estatística para estabelecer qual deles tem melhor comportamento.

3.3.1 Backtrack

Quando uma solução parcial S está sondada, nenhuma solução factível com valor melhor que o incumbente pode ser encontrada a partir de S desdobrando-se as variáveis restantes. No entanto, até que tal solução tenha sido encontrada algumas variáveis binárias foram fixadas, quer obrigatoriamente, como acontece em (c.2), quer por desdobramento. Caso uma ampliação tenha ocorrido por desdobramento, o complementar daquela variável deve ser analisado para que todas as 2^p combinações possíveis das variáveis binárias sejam implicitamente enumeradas. Por outro lado, uma ampliação cuja variável tenha sido fixada obrigatoriamente não requer a análise do complementar pois este é infactível. É conveniente, então, diferenciar as ampliações feitas obrigatoriamente para que este fato seja considerado. Isto é feito [14] colocando-se uma marca em S nas variáveis desdobradas deste modo.

No exemplo em (c.2), S passa de

$$S = \{1, -2\}$$

para

$$S = \{1, -2, 3, 4\}$$

O processo de análise das soluções complementares é denominado *Backtracking* e deve ser feito de forma sistemática para que nenhuma combinação deixe de ser enumerada. Uma forma de fazer isto é através da seguinte regra [14]:

> Localize o elemento mais à direita de S que ainda não está marcado. Se tal elemento não existir, fim do processo. Caso contrário, troque o elemento pelo seu complemento marcado e retire de S todos os elementos à sua direita.

Exemplificando um passo de *backtrack* em

$$S = \{1, -2, 3, 4\}$$

resultaria em

$$S = \{1, 2\}$$

Na realidade, o conjunto S é tratado como uma pilha, onde se busca o primeiro elemento a ser desdobrado, pois o primeiro elemento a ser retirado é o 4, que não pode ser desdobrado. Daí, prossegue-se retirando o 3, que também não o pode, até que se chegue ao 2.

3.3.2 Atualização do Incumbente

Quando $I^S = \emptyset$, tem-se que:

$$y_i^S = \sum_{j \in S} A_{i,j}\lambda_j + b_i \quad z < 0, i = 1, \ldots, n \tag{3.65}$$

$$y_i^S = \sum_{j \in S} A_{i,j}\lambda_j + b_i < 0, i = n+1, \ldots, n+m \tag{3.66}$$

Uma solução completa tem seus y_i^S's dados por:

$$y_i^S = \sum_{j \in S} A_{i,j}\lambda_j + b_i - \bar{z} + \sum_{j \notin S} A_{i,j}\lambda_j, i = 1, \ldots, n \tag{3.67}$$

$$y_i^S = \sum_{j \in S} A_{i,j}\lambda_j + b_i \sum_{j \notin S} A_{i,j}\lambda_j, i = n+1, \ldots, n+m \tag{3.68}$$

Observe que se

$$\sum_{j \notin S} A_{i,j}\lambda_j = 0 \tag{3.69}$$

Então os y_i^S's continuam indicando uma solução factível. Dado que uma maneira de garantir a Equação 3.69 é fazer $\lambda_j = 0, j \notin S$, esta solução é chamada *factível na completação com zeros*. Além disso, como sempre se busca uma solução com valor z menor que o incumbente, torna-se um novo incumbente que deve satisfazer 3.50. Então

$$z \quad \max_i\{b_i + \sum_{j \in S} A_{i,j}\lambda_j\}$$

Deve-se observar que o fato de uma solução factível ter sido encontrada com melhoria do incumbente não implica em sondagem da solução parcial S. Apenas o incumbente é atualizado, mas a busca de um ainda melhor deve continuar.

É importante observar, para efeito de convergência, que, após uma atualização de incumbente, pelo menos uma das restrições 3.53 está infactível pois $\bar{z}$ atual satisfaz

$$z < b_i + A_i\lambda,$$

para pelo menos um $i = 1, \ldots, n$. Isto implica na impossibilidade de haver uma atualização imediatamente após outra e, portanto, a convergência não é alterada pelas atualizações parciais. Este fato será utilizado a seguir para comprovar a convergência global do algoritmo.

3.3.3 Convergência

Um Fluxograma do algoritmo discutido agora é apresentado na Figura 3.10, que pode ser simplificado para o indicado na Figura 3.11. Agora observe-se a semelhança com o apresentado em [14], cuja convergência está provada(Figura 3.12). Dado que, como comentado na atualização do incumbente, é impossível haver uma atualização após a outra, a solução parcial corrente: ou será infactível, ou será desdobrada. Em qualquer dos casos incorre-se em uma operação de alteração na pilha S. Como tal número de operações é finito, conforme provado na referência supracitada, o algoritmo aqui apresentado converge.

3.3.4 Modificação

Na prática não se faz a busca de uma variável real com uma precisão infinita, mesmo porque não existem computadores capazes de atingir tal precisão. Portanto, estipula-se uma tolerância $\varepsilon > 0$, em torno do qual o valor ótimo z^0 deve estar:

$$z^0 = z - \varepsilon$$

Tomando-se uma tolerância proporcional a z tem-se:

$$z^0 = z - z\tau = z(1 - \tau)$$

Feito isto as desigualdades estritas das restrições 3.53 são substituidas por desigualdades simples; pois ao invés de $z < z$ busca-se $z < z - \varepsilon = z(1 - \tau)$.

$$z(1 - \tau) > z - b_i + A_i\lambda, i = 1, \ldots, n$$

ou, equivalentemente,

$$0 \geq [b_i - z(1 - \tau)] + A_i\lambda, i = 1, \ldots, n$$

Fazendo-se

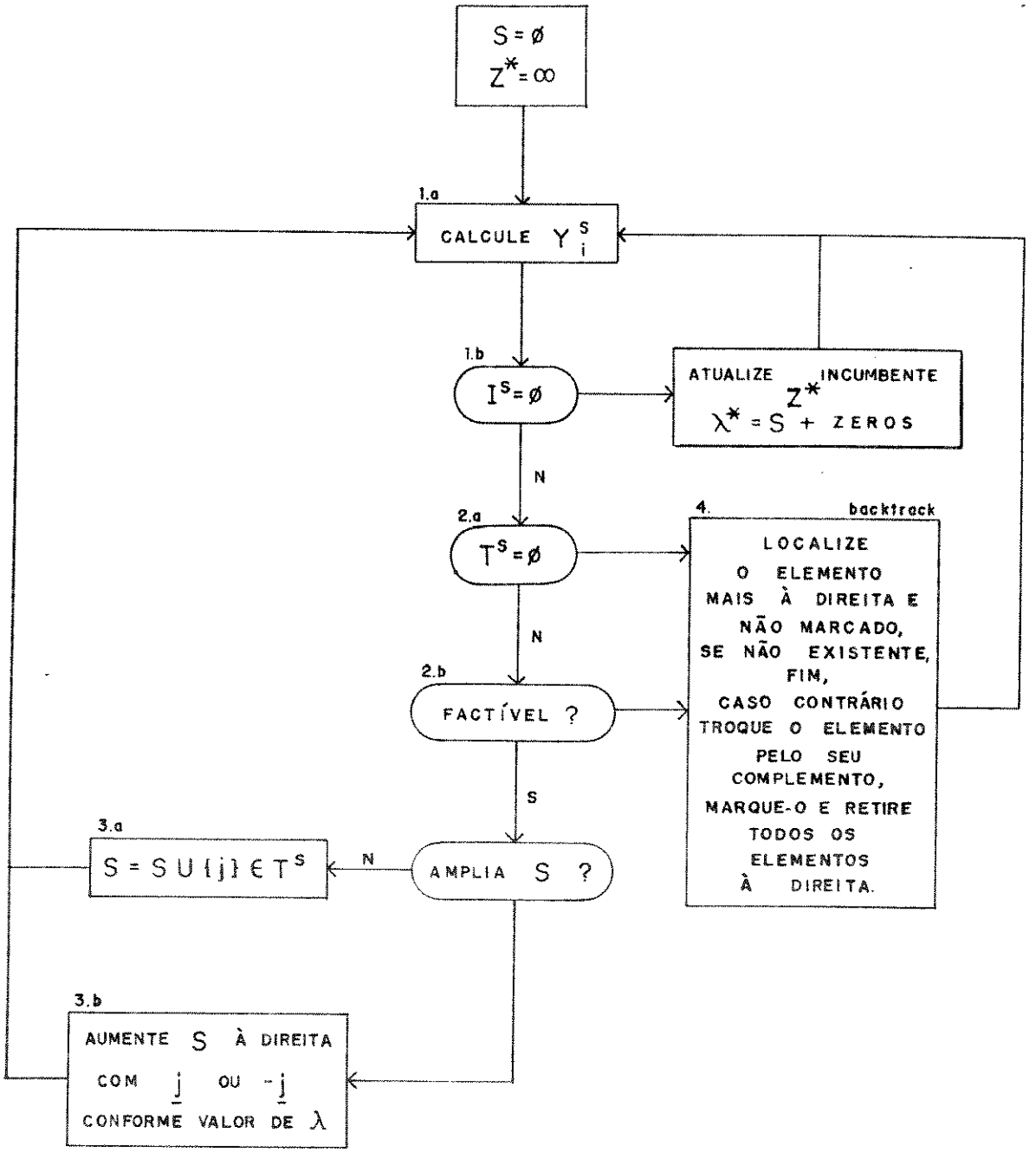

Figura 3.10.: Fluxograma do algoritmo.

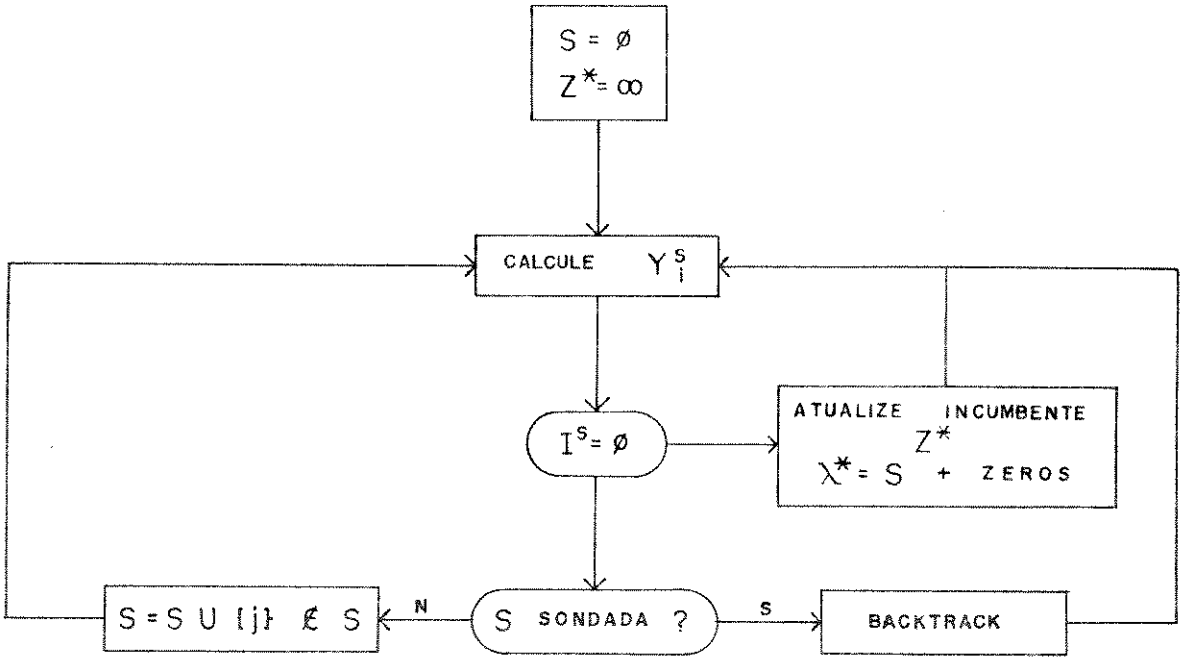

Figura 3.11.: Fluxograma simplificado.

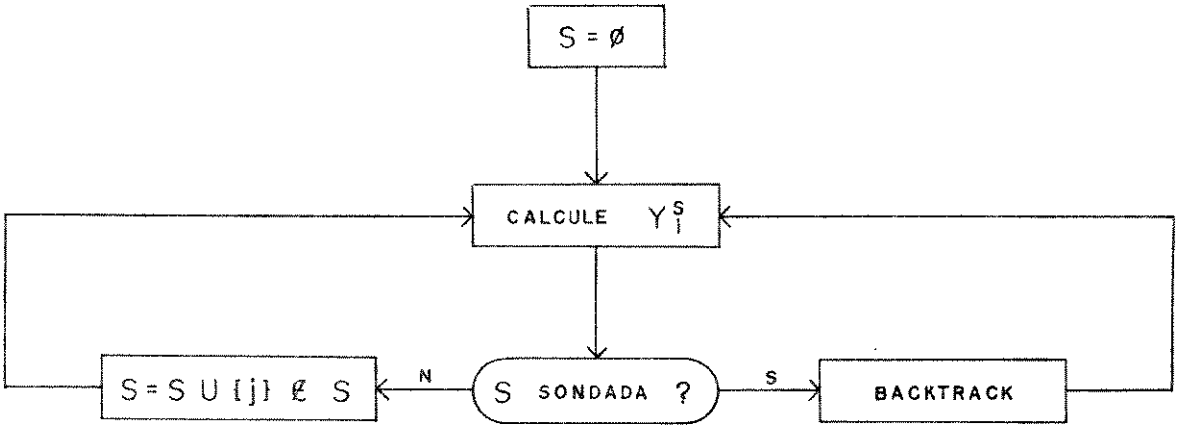

Figura 3.12.: Fluxograma apresentado por Geoffrion.

$$b'_i = b_i \quad z(1 \quad \tau) \quad , i = 1, \ldots, n$$
$$b'_i \qquad b_i \qquad\qquad\qquad , i \quad n + 1, \ldots, n + m$$

as desigualdades ficam semelhantes àquelas com $i = n + 1, \ldots, n + m$ e, portanto, podem ser tratadas da mesma forma, passando y_i^S e I^S a serem definidos como:

$$y_i^S = \sum_{j \in S} A_{i,j}\lambda_i + b'_i$$

$$I^S = \{i / y_i^S > 0, \qquad i = n + 1, \ldots, n + m\}$$

e os testes de sondagem e ampliação, simplificados para:

$$y_i^S + \sum_{j \in T^S} \min(0, A_{i,j}) > 0, \text{ para algum } i = 1, \ldots, n + m / y_i^S > 0$$

$$y_i^S + \sum_{j \notin S} \min(0, A_{i,j}) + | A_{i,k} | > 0, \qquad i \quad 1, \ldots, n + m, k \notin S$$

Porém, as atualizações do incumbente continuam sendo feitas apenas sobre as restrições com $i \quad 1, \ldots, n$, pois somente elas possuem a variável z.

3.3.5 Comentários Adicionais

- Não é difícil verificar que, sendo um algoritmo que procura factibilizar um incumbente melhor que o atual, pode-se utilizar o limitante superior dado pela melhor solução primal como limitante de partida. Este limitante é, obviamente, melhor que o $z = \infty$ utilizado como ponto de partida por levar a um aprofundamento mais rápido da solução parcial, além de indicar rapidamente se o método de Benders convergiu, sendo este último caso detectado quando não houver solução factível.

- No algoritmo modificado, utilizando-se a aproximação

$$z^0 \quad z \quad \varepsilon,$$

com $z \quad v_p$,a não existência de solução factível indica que a solução primal incumbente v_p é ε ótima.

- Também é possível adaptar o algoritmo para retornar logo após a primeira solução factível com $z = v_p$ ter sido encontrada, tornando possível sua utilização com a estratégia descrita em [13], comentada na seção 3.3 e ilustrando a versatilidade do algoritmo aqui proposto.

- A versão implementada foi construída de forma a testar tanto a versão modificada quanto a primeira. Desta forma, busca-se um valor de z satisfazendo a $z \leq \bar{z}(1-\tau)$ quando se deseja estabelecer um critério de parada com tolerância percentual τ.

3.4 Problema Mestre Dual

Como comentado em 3, a forma implementada da Decomposição Cruzada não utiliza o problema mestre dual (MD). Ao invés disto utiliza-se otimização por subgradientes para guiar a escolha dos multiplicadores duais C_t, $t \in T$. Nesta otimização usou-se o método de subgradiente modificado proposto por Camerini, Fratta e Maffioli citeCame, e Crowder [29]. Este método usa subgradientes prévios, exponencialmente suavizados, na correção da direção atual. Especificamente, se adota o seguinte esquema:

$$\begin{cases} C_t^0 & 0, \\ C_t^{m+1} & C_t^m + w_m s^m, \end{cases} \qquad t \in T \tag{3.70}$$

Neste esquema, w_m é o tamanho do passo a ser percorrido na direção s^m que é calculada através da recursão:

$$s^m = \mu^m + \beta_m s^{m-1}, \tag{3.71}$$

onde μ^m é um subgradiente da função dual e β_m é um escalar especialmente escolhido de forma a garantir que os multiplicadores duais C_t estejam cada vez mais próximos de seu valor ótimo C_t^*, isto é,

$$\| C_t^* - C_t^m \| > \| C_t^* - C_t^{m+1} \| \tag{3.72}$$

Deste modo, tende-se a eliminar o zig-zag característico do subgradiente puro, e as direções corrigidas são, pelo menos, tão boas quanto o próprio subgradiente.

Capítulo 4

Testes Computacionais

4.1 Geração dos Testes Computacionais

Não se conhece, da literatura, nenhum conjunto de dados com os quais se tenha realizado testes computacionais sobre o problema (P). Então, a realização de tais testes é feita com problemas gerados aleatoriamente, conforme descrito a seguir.

Demanda Para cada item e a cada período, as demandas foram geradas uniformemente entre 50 e 150;

Custos de preparação por unidade Gerados uniformemente entre 10 e 20, para cada item;

Custos de estocagem por unidade Gerados uniformemente entre 1 e 2 para cada item e cada período;

Capacidades de produção De modo a eliminar a geração de subproblemas duais infactíveis, cada problema é gerado satisfazendo, para cada $t \in T$, a desigualdade:

$$\sum_{j=1}^{t} M_{i,j} \geq \sum_{j=1}^{t} d_{i,j}$$

Contudo, como a geração de subproblemas duais com capacidade ilimitada também é indesejada, por não se estar testando o problema na forma proposta, uma solução de compromisso é a geração de capacidades entre $d_{i,j}$ e $\sum_{j=1}^{t} d_{i,j}$, pois a capacidade, na média, factível e limitada;

Custos de produção por unidade Gerados uniformemente entre 20 e 40, para cada item e cada período;

Tempos de preparação e de produção Gerados uniformemente entre 10 e 50, e 0,5 e 1,5 respectivamente;

Quantidade de horas disponíveis (Cap_t $m_t + l_t$) Este é sempre o último parâmetro gerado, pois é neste que se ajusta quão comprimido vai estar o problema em relação ao máximo que poderia ser requerido a cada período:

$$\sum_i (s_i + M_{i,t} p_i)$$

Toma-se, então, Cap_t com valores entre 40 e 90 por cento deste valor para que os problemas gerados sejam realmente capacitados neste parâmetro. Note que isto não garante que todos os problemas gerados sejam factíveis;

Custo unitário por hora utilizada gerados entre 40 e 80 com distribuição uniforme.

Um exemplo de um problema construído obedecendo os critérios anteriores para 12 períodos e 5 itens é ilustrado nas Tabelas 4.1 a 4.9. Note que este problema é capacitado tanto na quantidade de horas disponíveis por período quanto na capacidade de produção em cada período, pois:

(a) Nenhum plano da forma

$$\lambda_{i,1} \quad 1$$

$$\lambda_{i,t} \quad 0, t \neq 1$$

é factível, porque, como se pode observar a partir dos dados, todos os produtos violam a condição de factibilidade, ou seja, nenhum deles satisfaz:

$$M_{i,1} p_i \quad \sum_t d_{i,t} p_i$$

(b) Nenhum plano em que se decida produzir no primeiro período e não produzir nos três períodos seguintes, isto é,

$$\lambda_{i,1} \quad 1, \lambda_{i,2} \quad \lambda_{i,3} \quad \lambda_{i,4} \quad 0,$$

com $\lambda_{i,t}$ quaisquer para $t \quad 4$ é factível porque

$$Cap_1 \quad \sum_i (\sum_{t \quad 1}^{4} d_{i,t} p_i + \lambda_{i,1} s_i)$$

produto	*período*											
	1	2	3	4	5	6	7	8	9	10	11	12
1	16	12	18	13	15	11	16	18	13	18	17	13
2	11	16	12	12	17	11	14	14	15	11	10	19
3	11	10	10	14	12	18	11	19	11	14	11	16
4	17	11	14	11	13	10	10	18	19	13	15	19
5	19	11	13	12	14	19	12	18	17	11	14	13

Tabela 4.1: Custos de preparação por unidade.

produto	*período*											
	1	2	3	4	5	6	7	8	9	10	11	12
1	1,7	1,6	1,3	1,4	1,6	1,7	1,4	1,9	1,8	1,6	1,6	1,9
2	1,0	1,0	1,2	1,6	1,6	1,8	1,1	1,0	1,1	1,0	1,3	1,6
3	1,6	1,5	1,9	1,8	1,3	1,2	1,5	1,7	1,8	1,3	1,1	1,6
4	1,0	1,6	1,0	1,3	1,6	1,0	1,9	1,7	1,8	1,0	1,7	1,8
5	1,4	1,3	1,7	1,1	1,5	1,7	1,9	1,7	1,5	1,5	1,0	1,7

Tabela 4.2: Custos de estocagem por unidade.

produto	*período*											
	1	2	3	4	5	6	7	8	9	10	11	12
1	21	39	37	31	39	27	21	28	21	37	31	34
2	21	33	36	34	34	23	21	38	37	34	21	22
3	33	31	37	26	25	31	32	26	28	22	22	31
4	20	33	27	26	26	20	35	28	36	30	22	30
5	20	39	28	38	30	20	38	35	31	37	22	27

Tabela 4.3: Custos de produção por unidade.

produto	*período*											
	1	2	3	4	5	6	7	8	9	10	11	12
1	127	67	97	97	69	91	124	54	107	52	135	138
2	75	100	97	134	88	112	68	73	60	96	56	103
3	146	92	108	50	123	87	120	70	56	50	99	131
4	99	77	76	59	67	58	89	133	79	119	53	128
5	84	141	59	96	120	102	105	57	131	85	112	148

Tabela 4.4: Demandas.

produto	*período*											
	1	2	3	4	5	6	7	8	9	10	11	12
1	436	1085	654	112	354	309	179	777	581	848	722	876
2	498	768	844	1022	942	572	743	498	1052	804	548	570
3	391	221	411	177	1111	316	941	772	632	452	825	819
4	498	228	665	401	310	362	537	918	840	643	241	297
5	187	1005	443	1144	1045	799	1027	940	1174	863	525	152

Tabela 4.5: Capacidades de produção em unidades por período.

período	1	2	3	4	5	6	7	8	9	10	11	12
Cap_t	1492	1061	1404	1040	1205	1110	1168	1480	1598	1772	1589	1119

Tabela 4.6: Quantidades de horas disponíveis por período.

período	1	2	3	4	5	6	7	8	9	10	11	12
custo	68	69	65	64	51	41	67	46	68	73	79	52

Tabela 4.7: Custo unitário por hora utilizada em cada período.

produto	1	2	3	4	5
tempo	33	33	14	10	17

Tabela 4.8: Tempos de preparação por produto.

produto	1	2	3	4	5
custo	0,5	0,6	0,8	1,4	0,6

Tabela 4.9: Tempos gastos na produção de cada produto.

A programação dos algoritmos foi feita em linguagem C.

Na Tabela 4.10 são indicados os tempos de UCP em computador VAX, com a ordem de resolução seguindo da esquerda para a direita e de cima para baixo. Após a última iteração em (SD) foi detectado que uma solução abaixo de 1% da otimalidade fora encontrada (com um valor de 402.600) , satisfazendo um critério de parada pré-estabelecido. Note, no entanto, que estes tempos de UCP não representam um comportamento padrão do algoritmo, pois quando é utilizado um outro conjunto de dados obtém-se os resultados da Tabela 4.11.

O plano de produção $X_{i,t}$ e a quantidade de horas consumidas são ilustrados nas Tabelas 4.12 e 4.13.

4.2 Desempenho do Algoritmo

Com o objetivo de avaliar o desempenho do algoritmo de Decomposição Cruzada, utilizou-se os critérios de geração de dados descrito anteriormente para a criação de 45 problemas. A característica selecionada para esta avaliação é o *gap da solução*, ou seja, a diferença percentual entre os valores incumbentes primal e dual. Quando esta diferença é pequena, o algoritmo encontrou uma solução próxima do valor ótimo. Um sumário dos resultados obtidos é apresentado nas Tabelas 4.14 a 4.22. Nestas tabelas adotou-se a seguinte convenção:

problema número do problema;

$(a \times b)$ indica que o problema tem a variáveis inteiras e b variáveis reais;

N número de itens;

T número de períodos;

problema	(SD)	(SP)	(MD)	(MP)
tempo (s)	5,46	0,77		
	5,35	0,65		
	5,40	0,69		
	5,38			12,93
		0,67		
	5,54	0,65	93,88	
	5,38	0,66		
	5,59	0,64		
	5,53			

Tabela 4.10: Tempos de CPU em cada iteração.

problema	(SD)	(SP)	(MD)	(MP)
tempo (s)	7,38	0,78		
	7,27	0,76		
	7,64	0,84		
	7,40	0,70		
	7,52			198,26

Tabela 4.11: Tempos de CPU em cada iteração para um outro conjunto de dados.

produto	*periodo*											
	1	2	3	4	5	6	7	8	9	10	11	12
1	436	0	0	0	21	215	0	482	0	0	0	0
2	494	0	0	0	0	568	0	0	0	0	0	0
3	346	0	0	50	330	0	0	406	0	0	0	0
4	311	0	0	0	125	89	0	512	0	0	0	0
5	187	38	155	0	120	740	0	0	0	0	0	0

(*) produção no limite.

Tabela 4.12: Plano de produção obtido.

período	1	2	3	4	5	6	7	8	9	10	11	12
horas	1445,8	39,8	110	54	595,4	1110	0	1339,6	0	0	0	0

(+) no limite da quantidade disponível.

Tabela 4.13: Quantidades de horas utilizadas.

30×72								
problema	N	T	*tempo (s)*	*% ótimo*	(SP)	(SD)	(MP)	(MD)
1	5	6	83,65	< 1	21	21	6	5
2	5	6	11,33	< 1	4	4	1	
3	5	6	114,60	< 1	25	25	8	6
4	5	6	9,74	< 1	3	4		
5	5	6	81,70	< 1	25	25	7	4

Tabela 4.14: Conjunto de problemas 1 de 9.

tempo tempo computacional gasto na observação do problema. Note que quando a distância entre os incumbentes é menor que 1% o critério de parada é satisfeito e o tempo de observação é o gasto até que isto ocorra;

% ótimo indica a distância percentual entre os incumbentes após o tempo de observação. Quando esta distância é maior que 1% indica-se, ao lado, o tempo gasto até que esta é alcançada. Por exemplo, na Tabela 4.15, o problema 5 foi observado durante $884,21s$, sendo que nos primeiros $71,17s$ já havia chegado a uma distância de 3,65% entre os incumbentes, e em 360, isto é, $288,83s$ após, apenas conseguira reduzir esta distância a 2,14%. No tempo restante, até atingir os $884,21s$ não houve nenhuma melhoria dos destes incumbentes;

(SP) número de iterações do subproblema primal;

(SD) número de iterações do subproblema dual;

(MP) número de iterações do problema mestre primal;

(MD) número de maximizações da função dual feita através de subgradientes;

Analisando-se as Tabelas 4.14 a 4.22 é possível extrair as seguintes conclusões:

60×132								
problema	N	T	*tempo (s)*	*% ótimo*	(SP)	(SD)	(MP)	(MD)
1	10	6	238,55	< 1	4	5	1	
2	10	6	18,25	< 1	4	5	1	
3	10	6	878,90	$2,33$ $(< 107s)$	17	18	4	5
4	10	6	15,76	< 1	2	2		
5	10	6	884,21	$2,14$ $(< 360s)$	7	8	3	
				$3,65$ $(< 72s)$				

Tabela 4.15: Conjunto de problemas 2 de 9.

90×192								
problema	N	T	*tempo (s)*	*% ótimo*	(SP)	(SD)	(MP)	(MD)
1	15	6	48,10	< 1	7	8	1	1
2	15	6	31,06	< 1	6	7	2	
3	15	6	19,72	< 1	2	2		
4	15	6	900,00	$6,26$ $(< 50s)$	7	8	2	2
5	15	6	45,49	< 1	6	7	1	1

Tabela 4.16: Conjunto de problemas 3 de 9.

60×144								
problema	N	T	*tempo (s)*	*% ótimo*	(SP)	(SD)	(MP)	(MD)
1	5	12	21,41	< 1	2	2		
2	5	12	24,67	< 1	2	2		
3	5	12	900,00	$1,11$ $(< 20s)$	24	25	8	7
4	5	12	88,34	< 1	5	5	1	
5	5	12	900,00	$2,27$ $(< 50s)$	20	21	6	5

Tabela 4.17: Conjunto de problemas 4 de 9.

120×264								
problema	N	T	*tempo (s)*	*% ótimo*	(SP)	(SD)	(MP)	(MD)
1	10	12	900,00	4,68 ($< 50s$)	3	4	1	
2	10	12	39,14	< 1	2	2		
3	10	12	246,00	< 1	5	6	1	1
4	10	12	900,00	4,30 ($< 50s$)	4	5	1	
5	10	12	900,00	5,19 ($< 40s$)	3	4	1	

Tabela 4.18: Conjunto de problemas 5 de 9.

180×384								
problema	N	T	*tempo (s)*	*% ótimo*	(SP)	(SD)	(MP)	(MD)
1	15	12	50,77	< 1	2	2		
2	15	12	56,64	< 1	2	2		
3	15	12	900,00	2,10 ($< 60s$)	3	4	1	
4	15	12	900,00	3,59 ($< 90s$)	4	5	1	
5	15	12	49,43	< 1	3	3		

Tabela 4.19: Conjunto de problemas 6 de 9.

90×216								
problema	N	T	*tempo (s)*	*% ótimo*	(SP)	(SD)	(MP)	(MD)
1	5	18	74,61	< 1	2	2		
2	5	18	900,00	3,55 ($< 90s$)	8	8	2	2
3	5	18	92,00	< 1	2	2		
4	5	18	900,00	2,37 ($< 90s$)	8	8	2	2
5	5	18	900,00	2,76 ($< 140s$)	10	10	2	2

Tabela 4.20: Conjunto de problemas 7 de 9.

180 × 396								
problema	N	T	*tempo (s)*	*% ótimo*	(SP)	(SD)	(MP)	(MD)
1	10	18	354,27	< 1	4	5	1	
2	10	18	900,00	1,07	6	7	1	1
3	10	18	889,44	< 1 1,34($< 200s$)	5	5		1
4	10	18	137,34	< 1	2	2		
5	10	18	850,74	1,48 ($< 200s$)	3	4	1	

Tabela 4.21: Conjunto de problemas 8 de 9.

270 × 576								
problema	N	T	*tempo (s)*	*% ótimo*	(SP)	(SD)	(MP)	(MD)
1	15	18	1200,00	1,004 ($< 250s$)	3	4	1	
2	15	18	206,16	< 1	2	2		
3	15	18	1200,00	1,38 ($< 250s$)	3	4	1	
4	15	18	1200,00	1,97 ($< 250s$)	3	4	1	
5	15	18	1200,00	1,23 ($< 350s$)	4	5	1	

Tabela 4.22: Conjunto de problemas 9 de 9.

- *O desempenho da Decomposição Cruzada está intimamente ligado ao sucesso com o ping-pong.* Isto deve-se ao fato de que, em geral, os problemas mestres são muito mais difíceis de resolver que seus subproblemas correspondentes e , portanto, quando são muito solicitados, degradam o desempenho geral;

- *O crescimento do tempo computacional do subproblema dual com o número de períodos é não-linear, porém, com o número de itens, é linear.* Como pode ser visto analisando-se os problemas em que apenas os subproblemas primal e dual foram resolvidos. Nestes problemas, como o crescimento do tempo computacional do PFCM é linear com o número de arcos, apenas os subproblema duais tem influência na característica de crescimento não-linear detectado

  Com o número de itens a linearidade observada deve-se ao fato de que o subproblema dual, como já comentado, é separável em vários problemas menores, um para cada item, que são então resolvidos pelo algoritmo de programação dinâmica, no qual apenas o número de períodos influi na complexidade;

- *O crescimento do tempo computacional do problema mestre primal é muito forte (possivelmente exponencial) tanto com o número de períodos quanto com o de itens*, pois observando-se os problemas com 30, 60, 90 e 120 variáveis inteiras, o número máximo de problemas resolvidos foi 8,4,2 e 1;

- *O problema mestre dual, que foi substituido pela maximização da função dual através de subgradientes, segue a mesma característica do subproblema dual*, pois cada iteração daquele implica em um número fixo de iterações deste;

Capítulo 5

Conclusão

O método de Decomposição Cruzada é bastante atrativo e versátil. É atrativo porque explora as estruturas primal e dual de um problema de programação inteira mista, e é versátil no sentido de que não se precisa incluir ambos os problemas mestres, pois apenas um deles já é suficiente para garantir a convergência. Contudo, se apenas o problema mestre dual é utilizado, a convergência à otimalidade requer a inclusão de um algoritmo do tipo *desdobramento e sondagem* para fechar um eventual *gap* de dualidade. Em qualquer caso, ainda que não se resolva à otimalidade, o método fornece a qualquer iteração uma solução factível (se alguma existir) e uma medida da qualidade da solução dada pela avaliação de seus incumbentes primal e dual.

Experiências computacionais indicam que a eficiência do método depende fortemente do sucesso com o ping-pong, porque os problemas mestres são, em geral, de difícil resolução. Além disto, mesmo a utilização apenas do ping-pong apresenta um crescimento não linear com o número de períodos de tempo.

Pesquisas futuras podem ser realizadas explorando-se algumas variações que o algoritmo geral permite. São elas:

- A utilização, ao invés da maximização da função dual através de subgradientes, a resolução do problema mestre dual através de programação linear, mantendo-se o problema mestre primal ou retirando-o e incluindo-se um algoritmo de desdobramento e sondagem para fechar um eventual *gap* de dualidade;
- A resolução do subproblema dual através de um algoritmo de desdobramento e sondagem;
- A utilização de heurísticas, como as de Côté e Laughton [26], para a resolução do problema mestre primal, com o objetivo de tornar sua resolução mais eficiente;
- Testar a sensibilidade do problema com a capacidade de horas disponíveis;

Apêndice A

Sobre a Não-positividade da Variável Dual C_t

Será mostrado, a seguir que a variável C_t pode ser restrita a assumir apenas valores não-positivos.

Fixe um período t e suponha que $Y_{i,t} > 0$ para algum $i \in I$. Então, da Equação 3.3 em (SP_λ), segue-se que $R_t > 0$, ou $F_t > 0$, ou $R_t > 0$ e $F_t > 0$. Das Restrições 3.13 e 3.14 de (DSP) e de suas variáveis duais associadas, R_t e F_t, segue-se do teorema das folgas complementares que

$$R_t(-C_t + D_t - b_t)=0 \text{ e } F_t(\ C_t + E_t - q_t) = 0.$$

Conseqüentemente,

$$C_t = D_t - b_t \text{ ou } C_t = E_t - b_t\ .$$

Desde que $E_t \leq 0$ e $D_t \leq 0$ então $C_t \leq 0$.

Para um período fixo t e $Y_{i,t} = 0$ para qualquer $i \in I$ existem soluções duais múltiplas, e destas soluções é sempre possível escolher uma na qual $C_t \leq 0$:

- Se $\lambda_{i,t}s_i > 0$ para algum i, tem-se da Equação 3.3 que $R_t > 0$, ou $F_t > 0$, ou ainda $R_t > 0$ e $F_t > 0$, recaindo-se no caso anterior, implicando em $C_t < 0$;

- Se $\lambda_{i,t}s_i = 0$ para qualquer $i \in I$, $R_t = F_t = 0$, tem-se que todos os arcos ligando o nó correspodente a C_t como restante do grafo têm fluxo nulo conforme ilustrado na Figura A.1. Se numa dada solução básica este nó for ligado aos nós correspondentes a D_t ou E_t(Figura A.2), então, dado que $D_t \leq 0$ e $E_t \leq 0$ e que $C_t = D_t - b_t$ ou $C_t = E_t - q_t$, $C_t \leq 0$.

  Caso contrário, o nó estará conectado à arvore básica através de um dos $Y_{i,t}$, de acordo com a Figura A.3 e para que se possa garantir que $C_t \leq 0$, deve-se ligar seu nó

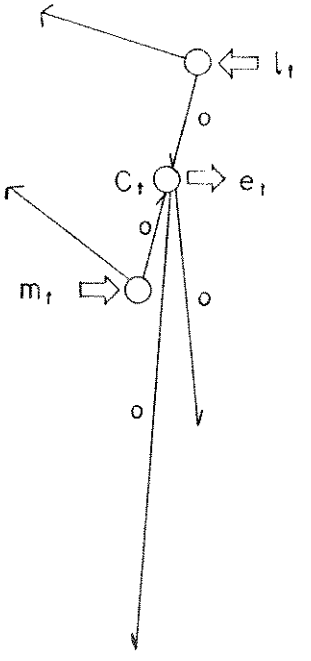

Figura A.1.: Trecho do grafo onde a variável C_t está inserida.

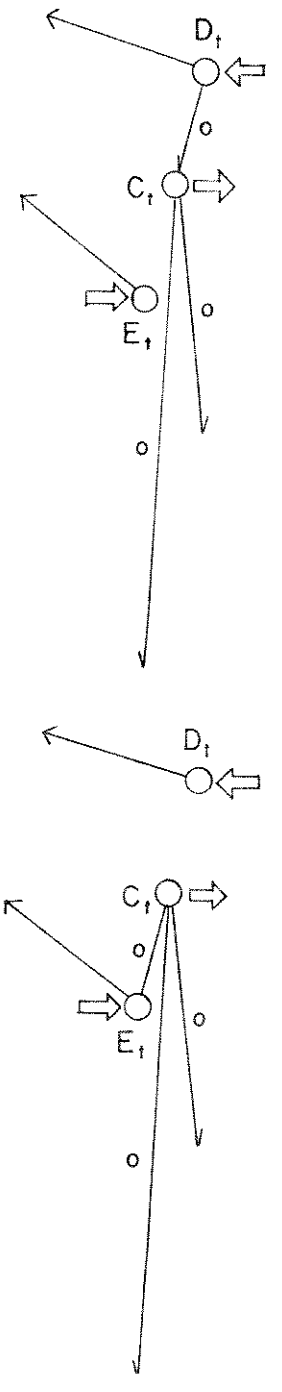

Figura A.2.: Configurações em que C_t se liga diretamente a D_t ou E_t.

associado aos nós correspondentes a D_l ou E_l. Esta ligação, ou seja, a colocação dos arcos R_l ou F_l na base formará um ciclo, descaracterizando-a. Porém caso algum outro arco do ciclo possa ser retirado a custo zero, a base continua ótima e está definida. Como, neste caso, há pelo menos um dos arcos $Y_{i,l}$ com fluxo zero nesta base, a troca pode ser efetuada. Além disto o custo da operação será nulo desde que não houve variação em nenhum dos fluxos da base.

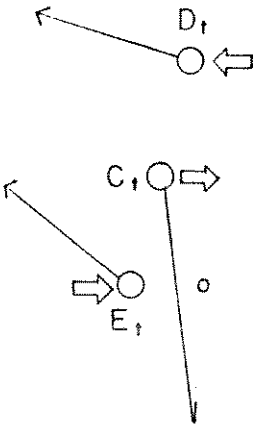

Figura A.3.: Configuração em que C_t está conexo à arvore através de $Y_{l,t}$.

Apêndice B

Estoques Finais

Se na última seqüência básica pelo menos um dos custos associados aos $Y_{i,t} > 0, t = T, T - 1, \ldots,$ ou pelo menos um dos custos de estocagem é estritamente positivo, o estoque final é nulo.

A prova desta afirmação é desenvolvida baseando-se em contradição das condições de otimalidade para o problema (SP_λ). Desde que o problema (P) poderia ser resolvido pela enumeração explícita de todas as combinações possíveis de $\lambda_{i,t}$, calculando-se o valor de (SP_λ), esta é uma propriedade do próprio problema (P).

Algumas das condições de otimalidade para (SP_λ) são, para cada $i \in I$:

$$H_{i,t}(-A_{i,t} + A_{i,t+1} - a_{i,t}/p_i) = 0, \qquad t = 1, \ldots, T-1, \tag{B.1}$$

$$H_{i,T}(A_{i,T} - a_{i,T}) = 0, \tag{B.2}$$

$$Y_{i,t}(A_{i,t} + B_{i,t} + C_t - v_{i,t}/p_i) = 0, \qquad t \in T, \tag{B.3}$$

$$B_{i,t}(Y_{i,t} - M_{i,t}p_i) = 0, \qquad t \in T. \tag{B.4}$$

Supondo $H_{i,t} > 0$, tem-se de B.2 que $A_{i,T} = a_{i,T}/p_i$. Como $H_{i,T-1} + Y_{i,T} - H_{i,T} = d_{i,t}p_i$, então $Y_{i,T} > 0$ ou $H_{i,T-1} > 0$ ou ainda, se $H_{i,T} > M_{i,T}p_i$ tem-se que $Y_{i,T} > 0$ e $H_{i,T-1} > 0$.

1. Se $Y_{i,T} > 0$

   (a) $M_{i,T}p_i > Y_{i,T} > 0$, tem-se de B.3 que $A_{i,T} + B_{i,T} + C_T - v_{i,T}/p_i = 0$ e de B.4 que $B_{i,T} = 0$. Então, $A_{i,T} - v_{i,T}/p_i - C_T$ é uma contradição se $a_{i,T} > 0$ ou se $v_{i,T}/p_i + C_T > 0$;

      i. $H_{i,T-1} = 0$ implica em fim da seqüência;

      ii. $H_{i,T-1} > 0$ a Equação B.1 implica em $A_{i,T-1} = A_{i,T} - a_{i,T}/p_i$, o processo é reiniciado com $H_{i,t-1}$;

(b) $Y_{i,T} = M_{i,T}p_i$, tem-se de B.3 $B_{i,T} = v_{i,T}/p_i - C_T - A_{i,T}$, que será positivo se $a_{i,T} > 0$ ou se $v_{i,T}/p_i - C_T > 0$, contradizendo $B_{i,T} \leq 0$;

i. $H_{i,T-1} = 0$ implica em fim da seqüência;

ii. $H_{i,T-1} > 0$ a Equação B.1 implica em $A_{i,T-1} = A_{i,T} - a_{i,T-1}/p_i$, e o processo é reiniciado com $H_{i,T-1}$;

2. Se $H_{i,T-1} > 0$ tem-se de B.1 que $A_{i,T-1} = A_{i,T} - a_{i,T-1}/p_i$, reiniciando o processo;

O processo continua recursivamente até que $H_{i,t} = 0$ para algum $t = T-1, T-2, \ldots$, ou seja, até que a última seqüência seja totalmente analisada. Então, se durante a busca nesta seqüência algum $a_{i,t} > 0$ ou algum $v_{i,t}/p_i - C_t > 0$ o problema tem suas condições de otimalidade violadas, pois antes de algum $H_{i,t} = 0$ existe, pelo menos, uma contradição nos valores de $B_{i,t}$ ou $A_{i,t}$. Na Figura B.1 estão ilustradas todas as possibilidades desta contradição.

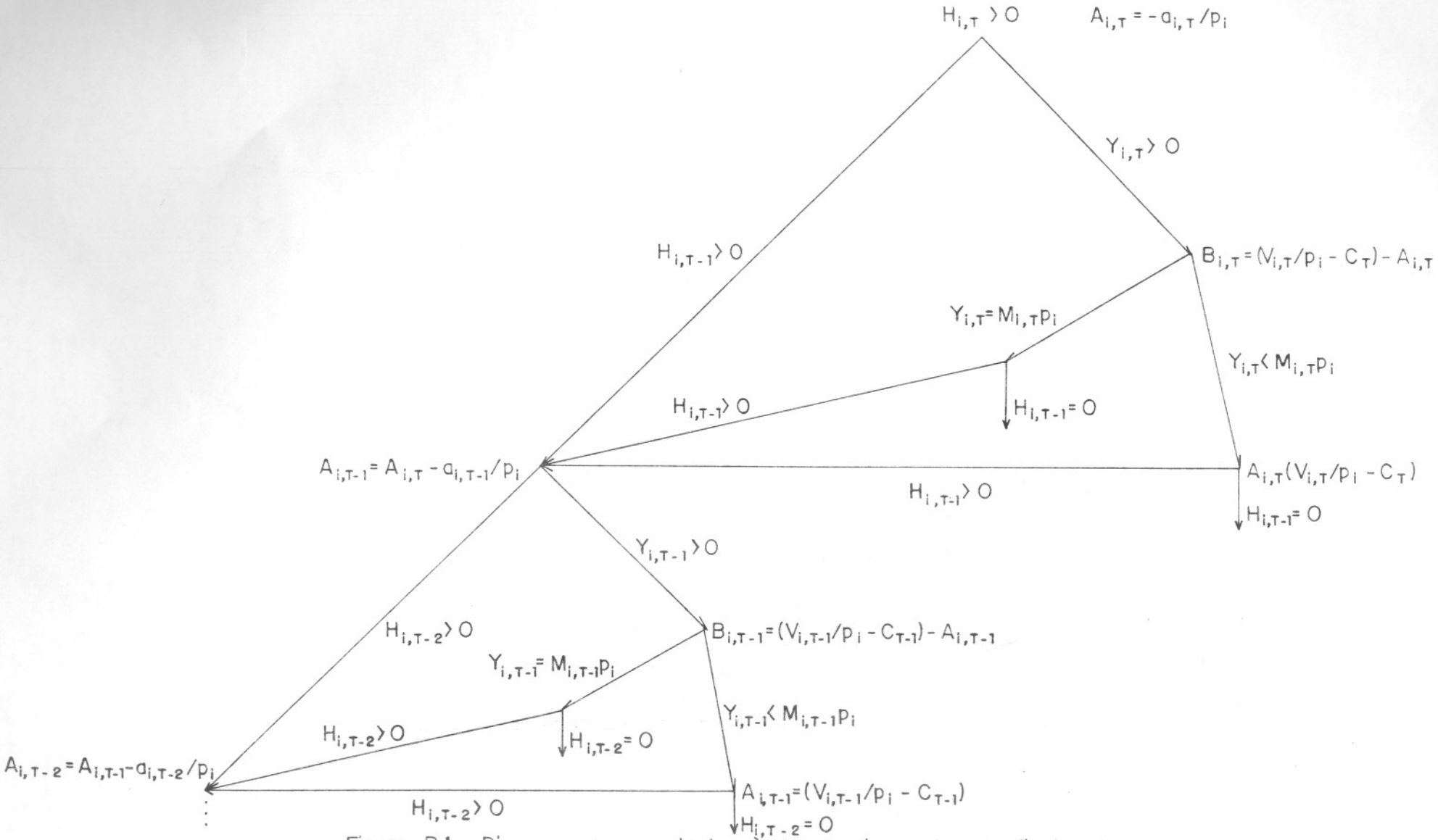

Figura B.1.: Diagrama correspondente à prova do estoque final nulo.

Bibliografia

[1] Wagner, H. M. e Whitin, T. M., (1958), *Dynamic Version of the Economic Lot Size Model*, Management Science 5(1), 89-96.

[2] Florian, M., Lenstra, J. K. e Rinnooy Kan, A. H. G., (1980), *Deterministic Production Planning: Algorithms and Complexity*, Management Science 26(7), 669-679.

[3] Evans, J. R., (1985), *Network-based Optimization Algorithms for the Capacitated Multi-item Lot Sizing Problem*, Comput. & Indus. Engng. 9(3),297-305.

[4] Bahl, H. C. e Zionts, S., (1987), *Multi-item schedulling by Benders' Decomposition*, Journal of the Operational Research Society 38(12), 1141-1148.

[5] Bahl, H. C., Ritzman, L. P. e Gupta, J. N. D., (1987), *Deterministic Lot Sizes and Resource Requirements: A review*, Operations Research 35(3), 329-345.

[6] Maes, J. e Van Wassenhove, L. N., (June 1986), *Multi Item Single Level Capacitated Dynamic Lotsizing Heuristics: A Computational Comparison (Part I: Static Case)*, IIE Transactions, 114-123.

[7] Karni, R., (1986), *A Study of Lot Sizing Algorithms*, IIE Transactions 18(4), 356-366.

[8] Newson, E. F. P., (1975), *Multi-item Lot Size Scheduling by Heuristic - Part 1: With Fixed Resources and Part 2: With Variable Resources*, Management Science 21, 1186-1203.

[9] Van Roy, T. J., (1983), *Cross Decomposition for Mixed Integer Programming*, Mathematical Programming 25, 46-63.

[10] Van Roy, T. J., (1986), *A Cross Decomposition Algorithm for Capacitated Facility Location*, Operations Research 34(1), 145-163.

[11] Gelders, L. F., Maes, J. e Van Wasenhove, L. N., *A Branch and Bound Algorithm for the Multi Item Single Level Capacitated Dynamic Lotsizing Problem*, Katholieke Universiteit Leuven, Division of Industrial Management, 92-108.

[12] Benders, J. F., (1962), *Partitioning for Solving Mixed Variables Programming Problems*, Mumerische Mathematike 4, 238-252.

[13] Geoffrion, A. M. e Graves, G. W., (1974), *Multicommodity Distribuition System Design by Benders Decomposition*, Management Science 20(5), 822-844.

[14] Geoffrion, A. M., (1967), *Integer Programming by Implicit Enumeration and Balas' Method*, SIAM Review 9(12), 178-190.

[15] Geoffrion, A. M. e Marsten, R. E., (1972), *Integer Programming Algorithms: A Framework and State-of-the-Art Survey*, Management Science 18(9), 465-491.

[16] Geoffrion A. M., (1974), *Lagrangean Relaxation for Integer Programming*, Mathematical Programming Study 2, 82-113.

[17] Lasdon, L. S., (1970), *Optimization Theory for Large Systems*, MacMillan, New York.

[18] Florian, M. e Klein, M., (1971), *Deterministic Production Planning with Concave Costs and Capacity Constraints*, Management Science 18(1), 12-20.

[19] Florian, M. e Robillard, P., (1971), *An Implicit Enumeration Algorithm for the Concave Cost Network Flow Problem*, Management Science 18(3), 184-193.

[20] Salkin, H. M., (1975), *Integer Programming*, Addison-Wesley Publishing Company.

[21] Trigeiro, W. W., Thomas, J. L. e McClain, J. O., (1987), *Capacitated Lot Sizing with Setup Times*, Working Paper No. 85-07, Johnson Graduate School of Management, Cornell University.

[22] Thizy, J. M. e Van Wassenhove, L. N., (1985), *Lagrangean Relaxation for the Multi-item Capacitated Lot-sizing Problem: A heuristic implementation*, IIE Transactions 17(4), 308-313.

[23] Baker, R. K., Dixson, P., Magazine, M. J. e Silver, E. A., (1978), *An algorithm for the Dynamic Lot-size Problem with Time-Varying Production Capacity Constraints*, Management Science 24(16), 1710-1720.

[24] Johnson, L. A. e Montgomery, D. C., (1974), *Operations Research in Production Planning*, Scheduling and Inventory Control, John Wiley, New York.

[25] Kennington, J. L. e Helganson, R. V., (1980), *Algorithms for Network Programming*, John Wiley, New York.

[26] Côté, G. e Laughton, M. A., (1984), *Large-scale Mixed Integer Programming: Benders-type Heuristics*, European Journal of Operational Research 16, 327-333.

[27] Balas, E., (1965), *An Additive Algorithm for Solving Linear Programs with Zero-one Variables*, Operations Research 13(4), 517-546.

[28] Camerini, P. N., Fratta, L. e Maffioli, F., (1975), *On Improving Relaxation Methods by Modified Gradient Techniques*, Mathematical Programmming Study 3, 26-34.

[29] Crowder, H., (1976), *Computational Improvements for Subgradients Optimization*, Symposia Mathematica 19, 357-372.