UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
DEPARTAMENTO DE ENGENHARIA DE SISTEMAS

Um Estudo Computacional da Busca Tabu Paramétrica para Programação Inteira Mista 0–1

Autor: Luís Henrique Sacchi
Orientador: Vinícius Amaral Armentano

Tese de Doutorado apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas - Unicamp, como parte dos requisitos para obtenção do título de Doutor em Engenharia Elétrica.
Área de concentração: Automação.

Banca Examinadora:
Prof. Dr. Cid Carvalho de Souza – UNICAMP
Prof. Dr. Eduardo Uchoa Barboza – UFF
Prof. Dr. Horacio Hideki Yanasse – INPE
Prof. Dr. Paulo Augusto Valente Ferreira – UNICAMP
Prof. Dr. Vinícius Amaral Armentano – UNICAMP (Orientador)

Campinas, SP
02/07/2010

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

Sacchi, Luís Henrique
Sa14e Um estudo computacional da busca tabu paramétrica para programação inteira mista 0-1 / Luís Henrique Sacchi. --Campinas, SP: [s.n.], 2010.

Orientador: Vinícius Amaral Armentano.
Tese de Doutorado - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Heurística. 2. Busca tabu. 3. Programação inteira. 4. Otimização combinatória. 5. Pesquisa operacional. I. Armentano, Vinícius Amaral. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Título em Inglês: A computational study of parametric tabu search for 0-1 mixed integer programs
Palavras-chave em Inglês: Heuristic, Tabu search, Integer programming, Combinatorial optimization, Operational research
Área de concentração: Automação
Titulação: Doutor em Engenharia Elétrica
Banca examinadora: Cid Carvalho de Souza, Eduardo Uchoa Barboza, Horacio Hideki Yanasse, Paulo Augusto Valente Ferreira
Data da defesa: 02/07/2010
Programa de Pós Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE DOUTORADO

Candidato: Luís Henrique Sacchi

Data da Defesa: 2 de julho de 2010

Título da Tese: "Um Estudo Computacional da Busca Tabu Paramétrica para Programação Inteira Mista 0-1"

Prof. Dr. Vinícius Amaral Armentano (Presidente): _____

Prof. Dr. Horácio Hideki Yanasse: _____

Prof. Dr. Eduardo Uchoa Barboza: _____

Prof. Dr. Cid Carvalho de Souza: _____

Prof. Dr. Paulo Augusto Valente Ferreira: _____

Resumo

Este trabalho apresenta um estudo computacional da busca tabu paramétrica para resolver problemas de programação inteira mista (PIM) com variáveis binárias. Trata-se de uma heurística genérica para problemas PIM gerais que resolve uma série de problemas de programação linear ao incorporar inequações de ramificação de variáveis inteiras como termos ponderados na função objetivo. O procedimento central do método é baseado em memória de curto prazo da busca tabu, enquanto fases de intensificação e diversificação são induzidas pela memória de longo prazo baseada em freqüência e idéias derivadas de *scatter search*. Novas estratégias são propostas para encontrar soluções de alta qualidade e extensivos testes computacionais são realizados em instâncias da literatura.

Palavras-chave: Programação Inteira, Meta-Heurística, Busca Tabu, Scatter Search, Memória Adaptativa, Ramificação Penalizada.

Abstract

We present a computational study of parametric tabu search for solving 0-1 mixed integer programming (MIP) problems, a generic heuristic for general MIP problems that solves a series of linear programming problems by incorporating branching inequalities as weighted terms in the objective function. The core procedure is founded on short term memory, whereas both intensification and diversification phases are induced by long term memory based on frequency and ideas derived from scatter search. New strategies are proposed for uncovering feasible and high-quality solutions and extensive computational tests are performed on instances from the literature.

Keywords: Integer Programming, Meta-Heuristic, Tabu Search, Scatter Search, Adaptive Memory, Penalized Branching.

Agradecimentos

Ao professor Vinícius Amaral Armentano pela oportunidade oferecida.

Aos meus pais a minha eterna gratidão.

À minha esposa Luciana pelo amor e incentivo.

Aos meus amigos da EEP: Martins e Camolesi pelas sugestões na apresentação.

À Sandra, à Célia, à Renata, ao Sérgio, ao Kleber, ao Guilherme e ao Martins por prestigiarem a apresentação.

Aos amigos que passaram pelo laboratório: André, Olinto, Gabi e Kazuo.

Aos amigos do laboratório: Luana, Yara, Ana Paula e Floriano pelo convívio.

À CAPES, pelo apoio financeiro.

Aos meus pais Luiz e Lourdes a minha gratidão

Dante and Virgil penetrating the forest, William Blake.*

Nel mezzo del cammin di nostra vita
mi ritrovai per una selva oscura,
ché la diritta via era smarrita.

Ahi quanto a dir qual era è cosa dura
esta selva selvaggia e aspra e forte
che nel pensier rinova la paura!

Tant'è amara che poco è più morte;
ma per trattar del ben ch'i' vi trovai,
dirò de l'altre cose ch'i' v'ho scorte.

Ao meio caminhar de nossa vida
fui me encontrar em uma selva escura
estava a reta minha via perdida.

Ah! que a tarefa de narrar é dura
essa selva selvagem, rude e forte
que volve o medo à mente que a figura.

De tão amarga, pouco mais lhe é a morte
mas, pra tratar do bem que enfim lá achei,
direi do mais que me guardava a sorte.

A divina comédia – Dante Alighieri - tradução Italo Eugenio Mauro - Ed. 34 - 1998.

*Impressão autorizada pelo museu Tate, `http://www.tate.org.uk`.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

$\lfloor v \rfloor$	-	Maior inteiro menor ou igual que v
$\lceil v \rceil$	-	Menor inteiro maior ou igual que v
$[v]$	-	Vizinho inteiro mais próximo de v
0^+	-	Desvio percentual inferior a 0,5%
ACVO	-	*Active-Constraint Variable Ordering*
CP_j	-	Valor de preferência de escolha na infactibilidade inteira
D	-	Conjunto das variáveis com infactibilidade inteira
d	-	Cardinalidade do conjunto D
D_0	-	Conjunto das variáveis com infactibilidade inteira que receberão metas
d_0	-	Cardinalidade do conjunto D_0
DM	-	Distância à meta
F	-	Conjunto de variáveis que assumem valores fracionários numa solução do PL'
f^+	-	Fração positiva $\lceil x''_j \rceil - x''_j$
f^-	-	Fração positiva $x''_j - \lfloor x''_j \rfloor$
GR_j	-	Valor da resistência a uma meta não atendida
GR_j^0	-	Valor da resistência potencial de uma infactibilidade potencial de meta
G	-	Conjunto das variáveis com infactibilidade de metas
G_p	-	Conjunto das variáveis com infactibilidade de metas que terão as metas invertidas
G_s	-	Conjunto das variáveis com infactibilidade de metas que terão as suas metas removidas
g_p	-	Cardinalidade do conjunto G_p
g_s	-	Cardinalidade do conjunto G_s
IP_j	-	Penalidade por não assumir um valor inteiro na infactibilidade inteira
M_j	-	Penalidade aplicada à variáveil x_j, é um número grande
N	-	Conjunto dos índices das variáveis 0–1 do vetor x, $N = \{1, \ldots, n\}$
N^+	-	Conjunto de variáveis que recebem metas (UP)
N^-	-	Conjunto de variáveis que recebem metas (DN)
N'	-	União dos conjuntos N^+ e N^-
n	-	Dimensão do vetor solução x
R-UP	-	Resposta (UP) a uma infactibilidade de meta (DN)
R-DN	-	Resposta (DN) a uma infactibilidade de meta (UP)
R-FREE	-	Resposta FREE, remover a meta de uma variável com infactibilidade de meta

T-UP - Transição (UP)
T-DN - Transição (DN)
T-FREE - Transição (FREE)
R-DN^D - Resposta (DN) na infactibilidade inteira
R-UP^D - Resposta (UP) na infactibilidade inteira
RB - *Reliability Branching*
RC_j - Custo reduzido na solução ótima do PL' da variável x_j
V-DN - Violação (DN)
V-UP - Violação (UP)
V-DN^0 - Violação (DN) de uma variável com infactibilidade potencial de meta
V-UP^0 - Violação (UP) de uma variável com infactibilidade potencial de meta
x - Vetor solução
x_j - Variável binária 0–1
x'_j - Valor da meta que a variável x_j deve atingir por meio de penalidades
x''_j - Solução encontrada pela relaxação linear PL' para a variável x_j
x_0^* - Melhor valor de função objetivo encontrado pela busca tabu paramétrica

Parâmetros da busca tabu paramétrica

ε - Usado na restrição de função objetivo
r - Usado no decaimento exponencial de M_j
NI - Usado no decaimento exponencial de M_j
T_0 - Controla variáveis potencialmente infactíveis com relação a metas
w - Usado na medida de preferência de escolha
μ - Usado na medida de preferência de escolha
k_1 - Controa a memória tabu
k_2 - Controa a memória tabu
k_3 - Controa a memória tabu
MaxLimpaTabuIter - Controa a memória tabu
η_{rel} - Usado no *Reliability Branching*
λ - Usado no *Reliability Branching*
γ - Usado no *Reliability Branching*
f_p - Controla as cardinalidades estáticas dos conjuntos G_p, G_s e D_0
f_s - Controla as cardinalidades estáticas dos conjuntos G_p, G_s e D_0
f_d - Controla as cardinalidades estáticas dos conjuntos G_p, G_s e D_0
$d_{\min}$ - Controla as cardinalidades estáticas dos conjuntos G_p, G_s e D_0
$g_{p_{\min}}$ - Controla as cardinalidades estáticas dos conjuntos G_p, G_s e D_0
$g_{s_{\min}}$ - Controla as cardinalidades estáticas dos conjuntos G_p, G_s e D_0
k_4 - Usado no controle adaptativo das cardinalidades dos conjuntos G_p, G_s e D_0
$g_{p_{\min}}$ - Usado no controle adaptativo das cardinalidades dos conjuntos G_p, G_s e D_0
$g_{s_{\min}}$ - Usado no controle adaptativo das cardinalidades dos conjuntos G_p, G_s e D_0
$f_{d_{\max}}$ - Usado no controle adaptativo das cardinalidades dos conjuntos G_p, G_s e D_0
δ_1 - Usado no controle adaptativo das cardinalidades dos conjuntos G_p, G_s e D_0

δ_2	-	Usado no controle adaptativo das cardinalidades dos conjuntos G_p, G_s e D_0
f_s	-	Usado no controle adaptativo das cardinalidades dos conjuntos G_p, G_s e D_0
$g_{\max}$	-	Usado no controle adaptativo das cardinalidades dos conjuntos G_p, G_s e D_0
Δss	-	Controla a relaxação na restrição de função objetivo
MaxRelax	-	Controla a relaxação na restrição de função objetivo
Δrm	-	Controla a relaxação na restrição de função objetivo
MaxCortes	-	Usado com cortes
MinDuracaoCortes	-	Usado com cortes
MultRR	-	Usado no B&C para resolver infactibilidade de metas
FracPermitida	-	Usado no B&C para resolver infactibilidade de metas
FracPorIter	-	Usado no B&C para resolver infactibilidade de metas
TempoRConf	-	Usado no B&C para resolver infactibilidade de metas
b	-	Controla o conjunto de referência
ω	-	Controla o conjunto de referência
ξ	-	Controla o conjunto de referência
α_cc	-	Controla as fases da busca com matriz de freqüência
α_int	-	Controla as fases da busca com matriz de freqüência
α_div	-	Controla as fases da busca com matriz de freqüência
k	-	Controla a geração de centros
f	-	Controla a geração de centros
V_{min}	-	Controla a geração de centros
V_{max}	-	Controla a geração de centros
maxRk	-	Usado na exploração de variáveis consistentes e fortemente determinadas
nivelMax	-	Usado na exploração de variáveis consistentes e fortemente determinadas
fracaoLimite	-	Usado na exploração de variáveis consistentes e fortemente determinadas
maxAnalisar	-	Usado na exploração de variáveis consistentes e fortemente determinadas
maxFix	-	Usado na exploração de variáveis consistentes e fortemente determinadas
γ_2	-	Usado na exploração de variáveis consistentes e fortemente determinadas

Trabalhos Publicados Pelo Autor

1. Sacchi, L. H. and Armentano, V. A.: A Computational Study of Parametric Tabu Search for 0–1 Mixed Integer Programs, Computers and Operations Research, Article in Press, doi:10.1016/j.cor.2010.07.004.

2. Armentano, V. A. and Sacchi, L. H.: Parametric Tabu Search for 0–1 Mixed Integer Programming: A Computational Study, *Simpósio Brasileiro de Pesquisa Operacional* (XLII SBPO / TC/ST – Trabalhos Completos para Sessões Técnicas), Bento Gonçalves, RS, Brazil, 2010.

3. Armentano, V. A. and Sacchi, L. H.: An Implementation of Parametric Tabu Search for Mixed Integer Programs. *VIII Metaheuristic International Conference* (MIC–2009), Hamburg, Germany, 2009.

Introdução

Problemas de programação linear inteira existem nas mais diversas áreas. Planejamento e programação da produção, decisão da localização de facilidades, projeto de redes de telecomunicação, planejamento de geração e transmissão de energia elétrica, escalonamento de linhas aéreas e roteamento de veículos são alguns exemplos de cenários que demandam o uso da programação inteira para otimizar uma função objetivo sujeita a restrições de recursos escassos.

O problema de programação inteira é classificado como NP-difícil e então os algoritmos conhecidos para resolverem o seu caso mais geral são de complexidade exponencial. A despeito dessa limitação, uma combinação de diversos fatores propiciou o sucesso da aplicação de métodos computacionais para resolver problemas práticos de grande magnitude. Entre eles, não pode ser negligenciada a grande evolução no desempenho do hardware. Mas foi principalmente a intensa pesquisa científica em métodos exatos e heurísticos que favoreceu o amadurecimento de pacotes comerciais baseados em *branch-and-cut*. Tais pacotes são de propósito geral e atualmente são capazes de resolver instâncias de problemas com milhares de variáveis em questão de horas. O surgimento das meta-heurísticas incluindo Algoritmos Genéticos (Goldberg, 1989), Busca Tabu (Glover e Laguna, 1997c), *Simulated Annealing* (Laarhoven e Aarts, 1987), *Scatter Search* (Glover, 1997) e GRASP (Feo e Resende, 1995) permitiu o desenvolvimento de algoritmos especializados para diversas classes de problemas. As meta-heurísticas não são capazes de provar a otimalidade de uma solução, mas conseguem encontrar soluções de boa qualidade em muitas instâncias em que os pacotes comerciais, ou métodos exatos falham quando um tempo limite de execução é imposto.

Embora exista uma profusão de heurísticas especializadas em classes específicas de problemas de otimização combinatória, há poucas heurísticas genéricas propostas na literatura. Recentemente, o interesse pelo desenvolvimento de heurísticas genéricas que interagem com pacotes de resolução de problemas de programação inteira mista (PIM) tem crescido. Este enfoque integrado é muito promissor, uma vez que casa a habilidade das heurísticas de encontrar soluções factíveis com os mecanismos sofisticados de busca em árvore dos pacotes. A descoberta de boas soluções factíveis no estágio inicial da busca permite a eliminação de ramos de baixa qualidade na árvore de enumeração, acelerando a prova da otimalidade, além da vantagem de obter soluções de alta qualidade em um

curto espaço de tempo.

Este trabalho envolve a implementação e o estudo computacional da heurística busca tabu paramétrica para problemas de programação linear inteira 0–1 mista proposta em linhas gerais por Glover (2006). A heurística faz uso intensivo da programação linear ao incorporar inequações de ramificação penalizadas na função objetivo. A estrutura rígida da busca em árvore é substituída pela memória tabu, flexível e adaptativa. Para comprovar a eficiência do método, extensos resultados computacionais são apresentados para instâncias de programação inteira de diversos problemas.

O trabalho está estruturado em seis capítulos. No Capítulo 1 é apresentada uma revisão bibliográfica das heurísticas gerais propostas na literatura de otimização inteira. O Capítulo 2 apresenta os fundamentos da busca tabu paramétrica de acordo com Glover (2006) e propõe algumas variações das táticas aplicáveis à busca com memória de curto prazo. O Capítulo 3 trata da implementação de extensões como uso de cortes, fixação de variáveis por implicações lógicas e integração da heurística com pacotes de *branch-and-cut*. Além disso, várias técnicas de longo prazo baseadas em conjunto de referência são avaliadas, algumas delas derivadas de *scatter search*. Os Capítulos 4 e 5 apresentam resultados computacionais em instâncias da Miplib 2003 (Achterberg et al., 2006), problema da mochila multidimensional com e sem restrições de demanda e o problema de designação generalizada. O Capítulo 6 conclui o trabalho e aponta novas direções de pesquisa relacionadas ao tema.

Capítulo 1

Revisão Bibliográfica

Embora exista um grande número de heurísticas especializadas em diversas classes de problemas de otimização combinatória, há relativamente poucas heurísticas genéricas propostas na literatura. Do ponto de vista do uso em algoritmos de *branch-and-cut*, as heurísticas gerais podem ser classificadas em dois grupos. O primeiro grupo é o das heurísticas de nó. Essas heurísticas procuram uma solução em um nó da árvore de enumeração a partir apenas dos dados do problema, como por exemplo a informação das variáveis fracionárias daquele nó. Quando usadas em conjunto com algoritmos de *branch-and-cut*, são aplicadas no nó raiz ou em nós internos com o intuito de diminuir o tamanho da árvore de enumeração, pois ramos de baixa qualidade automaticamente são descartados. O segundo grupo, o das heurísticas de busca local, sempre funcionam como auxiliadoras de algoritmos de *branch-and-cut* e só podem ser aplicadas quando uma solução incumbente existe. As heurísticas de busca local costumam fazer uso intensivo de pacotes comerciais para encontrar soluções de melhor qualidade na vizinhança da solução incumbente.

Esta revisão é formada da compilação de vinte e duas heurísticas publicadas entre 1969 e 2008. Procurou-se apresentá-las agrupadas por idéias similares e por ordem cronológica quando possível. Em primeiro lugar, são discutidas as heurísticas de três fases com ponto interior, e na seqüência são tratadas as heurísticas de pivotamento, heurísticas baseadas em cortes de convexidade e heurísticas de busca local com uso de pacotes comerciais. A discussão é fechada com a heurística *Feasibility Pump* e suas variações. Para facilitar o desenvolvimento, a despeito da natureza de minimização ou maximização dos trabalhos ter sido respeitada, foi usado um único modelo de problema de programação inteira mista (PIM) geral que é representado por

$$(\text{PIM})\ x_0 = \min\backslash\max cx + dy \tag{1.1}$$

sujeito a

$$Ax + Dy \geq b \tag{1.2}$$

$$x_j \in \{0, 1\} \quad \forall j \in \mathcal{B} \tag{1.3}$$

$$x_j \geq 0, \text{inteira} \quad \forall j \in \mathcal{G} \tag{1.4}$$

$$y_j \geq 0 \quad \forall j \in \mathcal{C} \tag{1.5}$$

em que A é uma matriz $(m \times n)$ e D é uma matriz $(m \times p)$. O conjunto de índices das variáveis inteiras $\mathcal{N} = \{1, \ldots, n\}$ é particionado em $(\mathcal{B}, \mathcal{G})$. $\mathcal{B}$ é o conjunto de índices das variáveis 0–1, $\mathcal{G}$ é o das variáveis inteiras gerais e $\mathcal{C}$ é o conjunto de índices das variáveis contínuas. A relaxação de programação linear do PIM que considera todas as variáveis como sendo contínuas é representada por PL. As restrições das variáveis 0–1 em (1.3) são relaxadas em $0 \leq x_j \leq 1, \forall j \in \mathcal{B}$, e as restrições (1.4) são relaxadas em $x_j \geq 0, \forall j \in \mathcal{G}$. O vetor solução x é inteiro se (1.3) e (1.4) são satisfeitas. Uma solução factível para o PIM é uma solução que é PL-factível e inteira.

1.1 Heurísticas de três fases com ponto interior

As primeiras heurísticas genuinamente genéricas que surgiram são compostas de três fases e foram bastante exploradas na literatura. As restrições precisam estar na forma $Ax \leq b$ ou $Ax + Dy \leq b$, pois assumem a existência de regiões factíveis com interior. Em linhas gerais, a Fase 1 toma o ponto extremo ótimo da relaxação linear PL do PIM como origem de uma direção a ser traçada para o interior ou borda da região factível. A Fase 2, partindo do ponto extremo ótimo, caminha ao longo dessa direção e coleta soluções inteiras-factíveis na vizinhança. Soluções inteiras infactíveis podem receber um tratamento factibilizador. A Fase 3 é de melhoria e só pode ser aplicada caso haja sucesso nas fases anteriores.

Hillier (1969) propôs uma heurística baseada nesse princípio para maximização em problemas de programação inteira pura e o modelo é tratado com restrições no padrão $Ax \leq b$. Em cada uma das fases, diferentes decisões táticas podem ser adotadas, levando a variações significativas nos algoritmos.

Na Fase 1, dois pontos $x^{(1)}$ e $x^{(2)}$ pertencentes ao poliedro são identificados. O ponto $x^{(1)}$ é a solução ótima do PL e $x^{(2)}$ é encontrado a partir de uma modificação no PL que o torna mais restrito. Mantendo a mesma base de $x^{(1)}$, o lado direito de cada restrição ativa é subtraído de um valor tal que após reotimizar e arredondar a solução ótima desse PL apertado, encontra-se o ponto $x^{(2)}$. Esse ponto $x^{(2)}$ é sempre factível para o PL original e como é uma solução inteira, já serve como uma solução inteira-factível para o PIM. São propostos dois métodos para a obtenção de $x^{(2)}$ na Fase 1. O Método

1 de obtenção de $x^{(2)}$ substitui o lado direito b_i de cada restrição ativa por $b_i^{(2)} = b_i - \frac{1}{2}\sum_{j\in B}|a_{ij}|$ em que B é o conjunto dos índices das variáveis básicas da solução ótima do PL. O Método 2 normaliza o modelo ao dividir cada restrição i por $\sqrt{\sum_{j=1}^{n} a_{ij}^2}$ encontrando b'[1] e A'. Então adota $b_i^{'(2)} = b_i' - \frac{\sqrt{N_0}}{2}$, em que N_0 é o número de variáveis em B excluídas as variáveis que são de folga no problema original. Em problemas com restrições muito apertadas, a Fase 1 pode falhar. Além disso, a idéia de apertar o PL não se aplica a restrições de igualdade, pois assume a existência de pontos interiores para o PL.

A Fase 2 caminha pelo segmento de reta entre $x^{(1)}$ e $x^{(2)}$, $x = (1-\alpha)x^{(1)}+\alpha x^{(2)}$, $0 \leq \alpha \leq 1$, sempre arredonda para o inteiro mais próximo e pára quando uma solução factível é encontrada. Como $(x_j^{(2)} - x_j^{(1)})$ é a taxa com que x_j cresce em relação a α, são identificados os valores de α que definem o ponto médio entre dois inteiros adjacentes. Uma segunda forma de gerar α é com incrementos de 0,05 a cada passo, seguido de arredondamento. Caso a solução inteira encontrada seja infactível, uma busca factibilizadora incrementa ou decrementa uma variável em uma unidade até que não seja mais possível reduzir a medida de infactibilidade q definida como $q = \sum_{i=1}^{m}\left(\max\left(0, \sum_{j=1}^{n} a'_{ij}x_j - b'_i\right)\right)$ ou $q = \max_{i\in\{1,\ldots,m\}}\left(\sum_{j=1}^{n} a'_{ij}x_j - b'_i\right)$. Dentre todas as variáveis que podem diminuir o valor q, escolhe-se aquela que ofereça o maior ganho $p = -\Delta q$ ou $p = \frac{c_j \Delta x_j}{-\Delta q}$, levando assim a quatro possibilidades de combinar as medidas q e p. Se for possível factibilizar, encerra-se a Fase 2, caso contrário, passa para o próximo valor de α. Uma terceira abordagem da Fase 2, e que se mostrou mais eficiente, combina as duas anteriores: α é atualizado de acordo com a primeira proposta e a busca factibilizadora é aplicada a cada passo.

A Fase 3 trabalha sobre uma solução encontrada nas fases anteriores e duas abordagens foram propostas. Na primeira, tenta-se aumentar ou diminuir uma variável em uma unidade desde que melhore a função objetivo. Para cada variável, a taxa de incremento na função objetivo em relação ao maior decréscimo observado em uma variável de folga é computada. A variável que apresenta o maior valor é escolhida para ser atualizada e o processo se repete até que um ótimo local seja alcançado. A segunda abordagem avalia duas variáveis simultaneamente, e atualizações de mais de uma unidade são permitidas em uma das variáveis caso o ganho líquido na função objetivo seja positivo e o par de movimentos aplicados não produza infactibilidade.

Os resultados reportados envolvem 24 instâncias aleatoriamente geradas com 15 restrições e 15 variáveis, 8 aleatoriamente geradas com 15 a 30 restrições e 15 a 30 variáveis e 4 instâncias reais do problema de transporte com custo fixo (*fixed-charge transportation*). As combinações mais promissoras de métodos para as 3 fases foram 1-2-1, 2-2-1, 1-3-1 e 2-3-1.

Influenciados por Hillier (1969), Ibaraki et al. (1974) apresentaram uma variação da heurística de três fases para maximização em programação inteira mista. As restrições aparecem na forma $Ax + Dy \leq b$ e cada restrição $i \in \{1, \ldots m\}$ é normalizada de modo que $\sum_{j=1}^{n} a_{ij}^2 + \sum_{j=n+1}^{n+p} d_{ij}^2 = 1$.

[1] b'_i se torna a distância euclideana do hiperplano i em relação à origem.

Ao contrário de um segmento de reta, a Fase 1 estabelece uma trajetória T linear por partes (vários segmentos de reta) para dentro do poliedro, tendo início em $x^{(1)}$. Os pontos de transição da direção dessa trajetória são encontrados ao resolver o PL (1.6) em que λ é uma nova variável escalar, Z é uma constante e e é um vetor com todas as componentes iguais a 1. Essa constante Z inicia com o valor ótimo da relaxação linear do PIM no vértice $x^{(1)}$ e é continuamente diminuída. A cada vez que o valor Z é diminuído, um novo segmento de reta é incluído na trajetória T. O processo é repetido por *CUT1* iterações.

$$\begin{aligned} &\max \lambda \\ &Ax + Dy + \lambda\, e \leq b \\ &cx + dy = Z \\ &x \geq 0 \\ &y \geq 0 \\ &\lambda > 0 \end{aligned} \tag{1.6}$$

A Fase 2 examina um número máximo de *CUT2* pontos na trajetória T ou é interrompida caso *NINT2* soluções inteiras-factíveis sejam identificadas. Os pontos na trajetória são arredondados para o inteiro mais próximo e o PL residual é resolvido para essa designação de variáveis inteiras, encontrando assim, os valores ótimos das variáveis contínuas. Se o PL for infactível, uma busca factibilizadora é aplicada tal que cada variável inteira é incrementada ou decrementada em uma unidade e aquela que apresentar a menor soma de violações nos lados direitos das inequações é escolhida para ser atualizada. Note que como há variáveis contínuas, cada teste exige reotimização de um PL. Uma variável que é incrementada ou decrementada fica proibida de sofrer um movimento no sentido contrário. Se após *DEPTH2* iterações a solução não foi factibilizada, o processo factibilizador é interrompido.

A Fase 3 tenta melhorar a qualidade das soluções encontradas na Fase 2. A informação do custo reduzido é utilizada e uma variável inteira é eleita para ser incrementada ou decrementada em uma unidade, e o processo se repete. Se o movimento de melhoria leva a uma infactibilidade, a busca factibilizadora da Fase 2 também é aplicada na Fase 3.

Os resultados foram reportados para instâncias aleatoriamente geradas com até 400 variáveis inteiras. Também há resultados para instâncias reais de minimização de *makespan* e *flow-time* em *jobshop*, problemas gerais de planejamento de investimento, *fixed-charge transportation*, particionamento de grafos, planejamento de projetos, problemas de *blending* e *logic design*. O número de variáveis inteiras para essas instâncias varia de 2 a 126. Os resultados encontrados foram de boa qualidade. Em instâncias de programação inteira pura essa heurística mostrou-se competitiva com Hillier (1969), apresentando tempos computacionais maiores que foram compensados por resultados

um pouco melhores.

Trabalhando em problemas de programação linear inteira mista, Faaland e Hillier (1979), por meio de uma análise estatística, propuseram um terceiro método para determinar $x^{(2)}$ na heurística Hillier (1969), uma vez que os métodos 1 e 2 da Fase 1 costumam ser conservativos demais. Assim usam $b_i^{(2)} = b_i - \sqrt{\sum_{j=1}^{N_0} a_{ij}^2}$, considerando apenas as variáveis básicas, excluídas as de folga e ordenadas de 1 a N_0. Essa variação oferece um arredondamento factível com probabilidade próxima a um. Esse método de apertar o lado direito das restrições, juntamente com os outros dois propostos por Hillier (1969), baseiam-se nas restrições funcionais ativas em $x^{(1)}$. Como há situações em que outras restrições podem influenciar o caminho em direção ao interior, eles propuseram um caminho interior linear por partes que depende da solução do seguinte PL parametrizado:

$$\begin{aligned} &\max r \\ &Ax + Dy + \Delta\, r \leq b \\ &cx + dy = Z \\ &x \geq 0 \\ &y \geq 0 \\ &r > 0 \end{aligned} \tag{1.7}$$

No PL (1.7), r é uma variável real escalar, Z é um valor que é continuamente diminuído em relação ao valor ótimo do PL original, obtido por $x^{(1)}$. Cada uma das componentes Δ_i do vetor Δ podem ser $\Delta_i = \frac{1}{2}\sum_{j\in B} |a_{ij}|$, $\Delta_i = \frac{1}{2}\sqrt{\sum_{j=1}^{n} a_{ij}^2} \times \sqrt{N_0}$ ou $\Delta_i = \sqrt{\sum_{j=1}^{N_o} a_{ij}^2}$ dependendo de qual Método 1, 2 ou 3 foi adotado na Fase 1. Os resultados computacionais foram competitivos com Ibaraki et al. (1974).

Saltzman e Hillier (1992) propuseram uma outra heurística de três fases para maximização em programação inteira pura baseada no conceito de ponto no teto. Um ponto no teto em relação à região factível é uma solução inteira-factível cujo incremento e/ou decremento de uma unidade em cada uma das variáveis x_j, individualmente, torna a solução infactível. Cada ponto extremo da caixa convexa das soluções inteiras do PI é um ponto no teto em relação à região factível. Computacionalmente é custoso avaliar se um ponto satisfaz essa condição, então o conceito de 1-ponto no teto em relação a uma restrição é explorado, amparado no teorema que garante que há uma solução ótima que é 1-ponto no teto em relação a uma restrição funcional para uma região factível limitada e não-vazia. Uma solução é 1-ponto no teto em relação a uma restrição i, se existir pelo menos uma variável x_j que quando incrementada ou decrementada em uma unidade, viole a restrição i. A Fase 1 da heurística aplica o método simplex para encontrar a solução ótima da relaxação linear $x^{(1)}$. Além disso, o *tableau* ótimo fornece todos os raios extremos que formam o cone convexo com vértice em $x^{(1)}$, e que após normalizados produzem o conjunto de direções $\{d^1, d^2, \ldots\}$. A Fase 2 da heurística

encontra o hiperplano h que define uma face da região factível de forma que $h = \arg\min_i \sum_{k \in E_i} cd^k$ e E_i representa o conjunto dos $(n-1)$ raios extremos que emanam de $x^{(1)}$ e sobre os quais jaz o hiperplano da restrição i. Enquanto Hillier (1969) e Faaland e Hillier (1979) procuram um ponto no interior do poliedro como guia, a heurística de ponto no teto afasta-se de $x^{(1)}$ sobre um dos hiperplanos ativos em $x^{(1)}$ que apresenta a menor taxa de decréscimo na função objetivo ao caminhar pela direção $d = \sum_{k \in E_h} d^k$ conforme ilustrado na Figura 1.1.

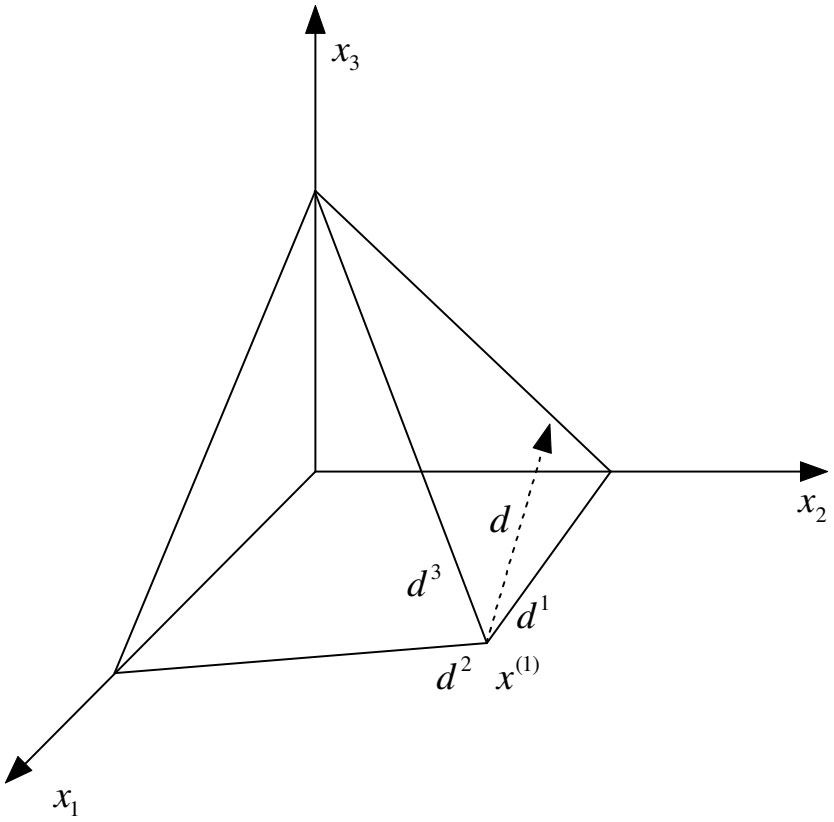

Fig. 1.1: Direção ao longo de uma face do poliedro.

Estabelecida a direção, move-se parametricamente a partir de $x^{(1)}$ ao longo do raio $x^{(1)} + \theta d$, determinando os valores positivos $\theta^1, \theta^2, \ldots$, tais que cada ponto $x^t = x^{(1)} + \theta^t d$ contém pelo menos uma componente inteira, situação em que as demais componentes são arredondadas pela expressão (1.8), gerando uma solução que é inteira-factível pelo menos em relação à restrição h.

$$x_j'^t = \begin{cases} \lfloor x_j^t \rfloor & \text{se} \quad a_{hj} > 0 \\ \lfloor x_j^t + \frac{1}{2} \rfloor & \text{se} \quad a_{hj} = 0 \\ \lceil x_j^t \rceil & \text{se} \quad a_{hj} < 0 \end{cases} \tag{1.8}$$

Esse mecanismo de arredondamento, nem sempre produz uma solução que é 1-ponto no teto em relação à restrição h, mas mesmo que isso não aconteça, a Fase 3 de melhoria garante que essa condição será satisfeita. A Fase 2 pode ser aplicada até que K_1 soluções tenham sido encontradas. Quando em uma direção d, a solução arredondada apresenta valor de função objetivo inferior a uma previamente encontrada para a mesma direção, um novo hiperplano h é escolhido. Se um ponto arredondado é um ponto extremo da caixa convexa das soluções inteiras, encerra-se a Fase 2. Se o ponto arredondado x' for infactível em relação a outras restrições que não h, e a soma das infactibilidades das restrições violadas não diminuir por K_2 iterações consecutivas, um novo hiperplano h é escolhido. Após um máximo de K_3 iterações, a Fase 2 é encerrada.

Há dois métodos aplicáveis na Fase 3 de melhoria. No Método 1, uma única variável é alterada. A componente de índice l é alterada em Δ_l sabendo que $l = \arg\max_j \{c_j \Delta_j\}$. Tal procedimento torna a solução encontrada 1-ponto no teto em relação à restrição que bloqueia maiores incrementos/decrementos em Δ_l. No segundo método aplicável na Fase 3, duas variáveis são alteradas. Primeiro uma variável x_j é incrementada ou decrementada em apenas uma unidade, procurando aumentar a soma das variáveis de folga. Depois o Método 1 de alterar uma única variável é aplicado. Eventualmente o incremento ou o decremento numa variável torna a solução infactível. Daí, a segunda variável modificada precisa factibilizar as restrições violadas e ainda fornecer o melhor ganho possível na função objetivo. A Fase 3 é aplicada iterativamente até atingir um ótimo local.

Embora essa heurística tenha apresentado resultados melhores que Balas e Martin (1980) descrita a seguir, Hillier (1969) supera essa heurística tanto em tempo quanto em qualidade. Embora a heurística possa ser aplicada a problemas 0–1, é mais recomendada para variáveis inteiras.

1.2 Heurísticas de pivotamento

Balas e Martin (1980) desenvolveram uma heurística para programação inteira 0–1 pura partindo da observação de que toda solução inteira-factível não possui variáveis 0–1 na base. As variáveis básicas devem ser as de folga do PL original enquanto que cada variável original $x_i, i \in \{1, \ldots, n\}$ está no limitante inferior ou superior, ou seja, 0 ou 1. O problema é apresentado na forma de maximização com restrições $Ax \leq b$.

Após resolver o PL na otimalidade, uma seqüência de pivotamentos é efetuada para inserir na base as variáveis de folga $s_i,\ i \in \{1, \ldots, m\}$ com custo mínimo na factibilidade dual e eliminando infactibilidades primais ao complementar algumas variáveis. Três tipos de pivotamentos são sugeridos. O Tipo 1 mantém a factibilidade primal: uma variável não-básica de folga entra e uma 0–1 sai da base. A variável escolhida para entrar na base é a que resulta no menor prejuízo para a função objetivo. O pivotamento Tipo 2 mantém a factibilidade primal e não altera o número de variáveis binárias na

base, ou seja, troca uma variável de folga por outra de folga ou uma binária por outra binária desde que reduza a medida de infactibilidade inteira computada pela expressão (1.9).

$$q = \sum_{i=1}^{n} \min(x_i, 1 - x_i) \tag{1.9}$$

O pivotamento do Tipo 3 remove uma variável 0–1 da base e insere uma de folga sacrificando a factibilidade primal. A variável de folga entrando na base deve assumir valor positivo. Arredondamentos e truncamentos são realizados na tentativa de encontrar uma solução inteira-factível. Novamente, o pivotamento que mais contribui com a redução da infactibilidade inteira calculada com a expressão (1.9) é o escolhido.

Caso nenhum dos três tipos de pivotamentos possa ser aplicado, tenta-se usar busca local. Uma solução vizinha da solução corrente é encontrada ao complementar exatamente uma variável 0–1. Todos os movimentos de complemento são avaliados, e o que apresenta a maior redução na expressão (1.9) é o escolhido, caso exista tal movimento de redução de infactibilidade. Essa técnica de escolher o melhor movimento de uma vizinhança é conhecida como *best improvement*. Caso não haja sucesso com essa vizinhança, adota-se uma vizinhança maior em que duas variáveis binárias são simultaneamente complementadas. Mas nesse contexto, o primeiro vizinho que provoca alguma redução na expressão (1.9) é imediatamente aceito. Essa técnica de escolher o primeiro movimento a apresentar melhoria para uma dada vizinhança é conhecida como *first improvement*. Se houver sucesso com o uso da busca local, a heurística volta a aplicar pivotamentos do Tipo 1.

Se uma solução inteira-factível for encontrada, uma busca local que complementa uma, duas ou até três variáveis, nessa ordem, é aplicada no final. Durante a busca local, no complemento de uma única variável, o *best improvement* é adotado, e no complemento de duas ou três variáveis, o *first improvement* é usado. Sempre que uma nova solução incumbente é encontrada, pela análise de custo reduzido da relaxação linear do problema original, o procedimento tenta fixar variáveis em 1 ou 0 e volta a tentar o complemento de uma variável.

A heurística foi testada em 92 instâncias de orçamento de capital (*capital budgeting*) com todos coeficientes positivos, além de instâncias de particionamento de conjuntos (*set partitioning*) e cobertura de conjuntos (*set covering*). Em apenas uma instância uma solução factível não foi encontrada, para 45 instâncias a solução ótima foi encontrada e o desvio ficou abaixo de 1% para 81 instâncias.

Balas et al. (2004) estenderam Balas e Martin (1980) para PIM gerais. As variáveis inteiras gerais são tratadas como 0–1 restritas ao intervalo entre os dois inteiros mais próximos do valor corrente da variável. O complemento de variáveis não-básicas 0–1 foi trocado pelo deslocamento de uma variável não-básica inteira do seu valor corrente a um vizinho inteiro. Aqui, os pivotamentos são ligeiramente diferentes, sempre preservam a factibilidade primal e são determinados pela regra da razão mínima

do método simplex. O pivotamento do Tipo 1 reduz $|I_1|$, o número de variáveis inteiras na base. Uma variável não-básica contínua é trocada por uma básica inteira. O pivotamento do Tipo 2 melhora a qualidade da função objetivo mantendo $|I_1|$ inalterado. Uma variável não-básica contínua é trocada por outra básica contínua ou uma inteira não-básica é trocada por outra inteira básica. O pivotamento Tipo 3 reduz a medida de infactibilidade inteira da solução corrente $\bar{x}$ computada pela expressão (1.10), mantendo $|I_1|$ inalterado.

$$q = \sum_{i \in I_1} \min \left(\bar{x}_i - \lfloor \bar{x}_i \rfloor , \lceil \bar{x}_i \rceil - \bar{x}_i \right) \tag{1.10}$$

Na fase de busca, os pivotamentos do Tipo 1 são aplicados à exaustão. Em seguida o arredondamento ao inteiro mais próximo é aplicado e interrompe-se a busca se a solução for factível. Caso contrário, pares de pivotamentos do Tipo 3 e Tipo 2 são aplicados e arredondados até que uma solução factível seja encontrada. Caso nenhum pivotamento do Tipo 3 possa ser aplicado, uma busca em uma vizinhança pequena é aplicada em torno de uma solução parcial. Tal solução parcial é criada com auxílio de (1.11) e é constituída das variáveis cujos valores estão muito próximos da integralidade.

$$S = \{ i \in I_1 | \min \{ \bar{x}_i - \lfloor \bar{x}_i \rfloor , \lceil \bar{x}_i \rceil - \bar{x}_i \} \leq \alpha \} \tag{1.11}$$

Em (1.11), α é tipicamente pequeno como $\alpha = 0,1$. O valor x_i^* é obtido pelo arredondamento de cada $\bar{x}_i$, $i \in S$ ao inteiro mais próximo. Então um par de restrições (1.12) é imposto e o PIM resultante é resolvido.

$$\sum_{j \in S} x_j^* - 1 \leq \sum_{j \in S} x_j \leq \sum_{j \in S} x_j^* + 1 \tag{1.12}$$

Se a fase de busca é bem sucedida, a fase de melhoria é aplicada. Variáveis 0–1 não-básicas podem ser deslocadas ao extremo oposto. As variáveis inteiras podem ser deslocadas uma unidade acima ou abaixo do seu valor atual. Deslocamentos duplos (em duas variáveis simultaneamente) e triplos podem ser aplicados, sempre adotando o *first improvement*. Quando essa busca atinge um mínimo local com a solução $\bar{x}^*$, uma busca numa vizinhança maior é executada. Um PIM é criado impondo as restrições (1.13) em que $\mathcal{N}$ é o conjunto de variáveis inteiras e $k > 1$.

$$\sum_{j \in \mathcal{N}} \bar{x}_j^* - k \leq \sum_{j \in \mathcal{N}} x_j \leq \sum_{j \in \mathcal{N}} \bar{x}_j^* + k \tag{1.13}$$

A heurística combinada com o Xpress 14.20 foi aplicada a 68 problemas da MIPLIB 3 (Bixby et al., 1998) e Mittelmann (2003) e comparada com o Xpress 14.20 sozinho. Com limite de tempo de 20 minutos, foi melhor em 39 casos.

Løkketangen et al. (1994) incrementaram o algoritmo de Balas e Martin (1980) com a adição de memória tabu. Pivotamentos do Tipo 1 e 2 são permitidos na fase de busca e movimentos que pioram a medida de infactibilidade inteira também o são. Uma lista tabu das últimas k variáveis que entraram na base é mantida. Na fase de melhoria, apenas o complemento de uma única variável é utilizado e somente movimentos que mantêm a factibilidade são aceitos como integrantes da vizinhança. São permitidos, porém, movimentos que pioram a qualidade do valor da função objetivo. As variáveis que mais recentemente foram complementadas são consideradas tabu. Foram utilizadas 57 instâncias de mochilas múltiplas disponíveis na OR-Library mantida por Beasley (1990). Os resultados computacionais são superiores a Balas e Martin (1980) com a contrapartida de tempos de execução ligeiramente maiores.

Considerando minimização em PIM 0–1, uma terceira variação da heurística de pivotamento foi apresentada por Eckstein e Nediak (2007). O método usa minimização de uma função côncava $\psi(.)$ que mede a infactibilidade inteira segundo a expressão (1.14). Diferentes valores do parâmetro α podem ser usados e $\mathcal{B}$ é o conjunto das variáveis binárias. A função ψ é definida por

$$\begin{aligned} \psi(x) =& \sum_{j \in \mathcal{B}} \phi_j \left(x_j\right) \\ \phi_\alpha(x) =& 1 - \begin{cases} \left(\frac{x-\alpha}{\alpha}\right)^2, & \text{se} \quad x \leq \alpha \\ \left(\frac{x-\alpha}{1-\alpha}\right)^2, & \text{se} \quad x > \alpha \end{cases} \end{aligned} \tag{1.14}$$

Partindo da solução ótima do PL, $x^{(1)}$, pivotamentos que diminuem o valor de $\psi(.)$ com prejuízo mínimo na função objetivo são realizados. Se um mínimo local é alcançado sem que seja uma solução inteira, uma fase de inspeção é iniciada. Os pontos extremos vizinhos ao vértice corrente são analisados em relação à expressão (1.15) até que um deles seja aceito. Em (1.15), x é o vértice corrente, x' é um vizinho e μ é um parâmetro da heurística. Nessa expressão, $\frac{cx'-cx}{\psi(x)-\psi(x')}$ representa a taxa de degradação na função objetivo em relação à melhoria na função $\psi(.)$ e $\frac{cx-cx^{(1)}}{\psi(x^{(1)})-\psi(x)}$ representa a degradação ocorrida desde o início da heurística.

$$\frac{cx' - cx}{\psi(x) - \psi(x')} \leq \mu \frac{cx - cx^{(1)}}{\psi(x^{(1)}) - \psi(x)} \tag{1.15}$$

Caso nenhum ponto x' atenda a expressão (1.15), aceita-se o ponto que apresente o menor valor de $\frac{cx'-cx}{\psi(x)-\psi(x')}$, com $\psi(x') < \psi(x)$. Se nenhum ponto vizinho satisfizer essas duas condições, a fase de inspeção falha. Recorre-se então a uma fase de corte, que adiciona um corte de convexidade (ou corte de intersecção), (Balas 1971, Balas 1972, Balas et al. 1971, Glover 1973) derivado da informação coletada na fase de inspeção. Neste passo da heurística pode-se voltar para a fase de pivotamento ou opcionalmente iniciar uma fase de busca em profundidade que fixa simultaneamente várias variáveis por arredondamento ao inteiro mais próximo. Se o subproblema com variáveis fixadas for factível,

reaplicam-se todos os passos da heurística recursivamente ao subproblema. Porém, se o subproblema com variáveis fixadas tornar-se infactível, aplica-se um corte canônico da expressão (1.16) (Balas e Jeroslow, 1972) ao problema original. Esse corte exclui a designação parcial de variáveis fixadas, observando que $\mathcal{B}_1$ é o conjunto das variáveis binárias fixadas em um no subproblema. Aplica-se reotimização, e caso o problema seja factível, volta-se para a fase de pivotamento. O uso de um corte canônico ou um corte de qualidade da função objetivo permite a continuidade da busca quando uma solução inteira-factível for encontrada.

$$\sum_{j \in \mathcal{B}_1} (1 - x_j) + \sum_{j \in \mathcal{B} \setminus \mathcal{B}_1} x_j \geq 1 \tag{1.16}$$

Os testes foram conduzidos em 42 instâncias da MIPLIB 3, o método não encontrou uma solução inteira-factível em 10 casos e encontrou 3 soluções ótimas. O desvio em relação à solução ótima foi inferior a 10% em 18 casos e acima de 100% em 4 casos.

Løkketangen e Glover (1998) desenvolveram uma busca tabu para PIM 0–1 baseada em pivotamentos aplicados ao PL. Nessa heurística, um movimento da busca é equivalente a um pivotamento aplicado ao *tableau* simplex. A vizinhança de um vértice $x(0)$ é formada pelos vértices vizinhos encontrados com a expressão (1.17). *NB* é o conjunto dos índices das variáveis não-básicas, D_h é o vetor coluna associado à variável não-básica x_h no *tableau* atual e θ_h é um escalar positivo que move a solução de $x(0)$ a $x(h)$ ao longo da aresta que os conecta. Quando um pivotamento é efetuado, a variável que deixa a base fica proibida de retornar durante as próximas t iterações, ou seja, é classificada como tabu.

$$x(h) = x(0) - D_h \theta_h \quad \forall h \in NB \tag{1.17}$$

Para cada movimento, a variação observada na qualidade da função objetivo é expressa por $\Delta z = z(h) - z(0)$, enquanto que a variação na medida de infactibilidade inteira é $\Delta u = u(h) - u(0)$. Para problemas de minimização, Δz negativo significa uma melhoria na qualidade. Já Δu negativo é sempre atrativo tanto para minimização quanto para maximização. A infactibilidade inteira é calculada com a expressão (1.18) na qual $[.]$ representa o arredondamento para o inteiro mais próximo, B é o conjunto dos índices das variáveis básicas e p é uma constante.

$$\begin{aligned} u_j(h) &= |x_j(h) - [x_j(h)]|^p \quad \forall j \in B \\ u(h) &= \sum_{j \in B} u_j(h) \end{aligned} \tag{1.18}$$

Os movimentos são agrupados nos conjuntos $H_1, \ldots, H_4$ de acordo com as regiões de I a IV na Figura 1.2 e são assim classificados.

- Tipo I diminui a infactibilidade inteira e piora a qualidade da função objetivo, ou seja, $\Delta z > 0$ e $\Delta u < 0$.
- Tipo II aumenta a infactibilidade inteira e melhora a qualidade da função objetivo, ou seja, $\Delta z < 0$ e $\Delta u > 0$.
- Tipo III diminui ou mantém a infactibilidade inteira e melhora ou mantém a qualidade da função objetivo, ou seja, $\Delta z \leq 0$ e $\Delta u \leq 0$.
- Tipo IV aumenta ou mantém a infactibilidade inteira e piora ou mantém a qualidade da função objetivo, ou seja, $\Delta z \geq 0$ e $\Delta u \geq 0$, exceto ($\Delta z = 0$ e $\Delta u = 0$) que é classificado como Tipo III.

Quatro regras diferentes de classificação dos movimentos foram propostas e são embasadas em diversas medidas de avaliação dos movimentos $E(h) \equiv f(\Delta z, \Delta u)$.

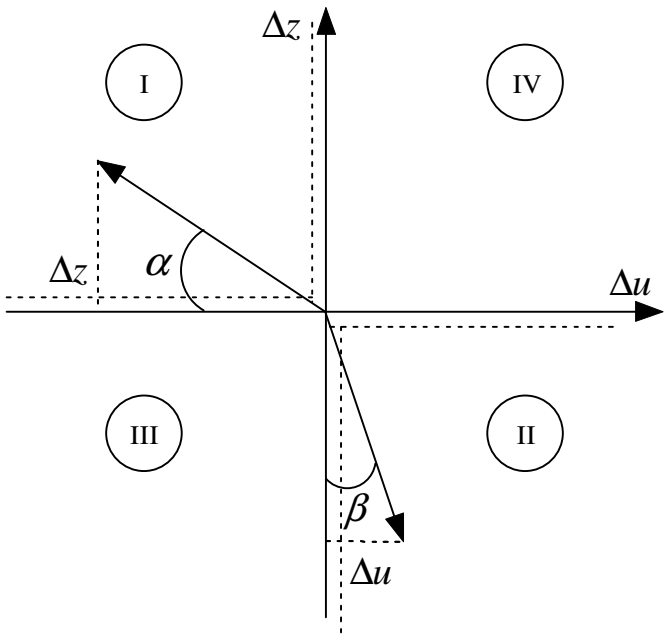

Fig. 1.2: Classificação de movimentos por quadrantes.

A primeira regra de escolha é a soma ponderada que usa a medida de avaliação (1.19), na qual w_1 e w_2 são constantes positivas. Quanto menor o valor em (1.19), mais atrativo é o movimento. A terceira regra de escolha proposta considera também a soma ponderada, mas privilegia os movimentos dos quadrantes I e III. Somente na ausência deles é que os movimentos nos quadrantes II e IV são considerados.

$$E(h) = w_1 \Delta z(h) + w_2 \Delta u(h) \tag{1.19}$$

A segunda regra de escolha é o teste da razão que sempre privilegia os movimentos do quadrante III. Na ausência desses movimentos, os do quadrante I e II são tratados simultaneamente e os do quadrante IV são os movimentos menos desejáveis. Portanto, somente são avaliados sob ausência dos demais, lembrando que o critério tabu exclui movimentos.

O cálculo de $E(h)$ para o movimento do quadrante I é por (1.20) e escolhe-se o movimento que ofereça o maior valor, minimizando assim o ângulo α ilustrado na Figura 1.2 e em caso de empate, aplica-se o movimento que apresente maior valor $|\Delta u|$.

$$E(h) = \frac{\Delta z(h)}{\Delta u(h)} \tag{1.20}$$

Para movimentos no quadrante II, a expressão usada é a (1.21), e escolhe-se o movimento que ofereça o maior valor, minimizando assim o ângulo β presente na Figura 1.2, e em caso de empate, aplica-se o movimento que apresente maior valor $|\Delta u|$.

$$E(h) = \frac{\Delta u(h)}{\Delta z(h)} \tag{1.21}$$

Os movimentos no quadrante III são avaliados segundo (1.22) e o movimento com maior valor é o escolhido. Caso haja empate, escolhe-se o movimento que apresente maior valor entre $\min(|\Delta z|, |\Delta u|)$. Caso $\Delta u(h) = 0$ para todos os movimentos, aceita-se o com maior valor $|\Delta z(h)|$, e similarmente, caso $\Delta z(h) = 0$ para todos os movimentos, aceita-se o com maior valor $|\Delta u(h)|$. Em casos de degenerescência, quando $\Delta z(h) = \Delta u(h) = 0$, aceita-se o movimento com uma probabilidade estabelecida de $1/3$ ou $1/10$, por exemplo.

$$E(h) = \Delta u(h) \times \Delta z(h) \tag{1.22}$$

Os movimentos no quadrante IV são também escolhidos por (1.22), mas neste caso o movimento com menor valor é o escolhido. Caso haja empate, escolhe-se o movimento que apresente menor valor entre $\min(|\Delta z(h)|, |\Delta u(h)|)$. Caso $\Delta u(h) = 0$ para todos os movimentos, aceita-se o com menor valor $|\Delta z(h)|$, e similarmente, caso $\Delta z(h) = 0$ para todos os movimentos, aceita-se o com menor valor $|\Delta u(h)|$.

No tratamento unificado dos movimentos dos quadrantes I e II, a vizinhança considerada é $H = H_1 \cup H_2$. Dois modelos de normalização foram propostos para aceitar h^*, o melhor movimento entre h_1 e h_2, que são os melhores movimentos de H_1 e H_2 computados por (1.20) e (1.21), respectivamente. A normalização 1 segue (1.23) em que q e w são parâmetros positivos a serem determinados.

$$
\begin{aligned}
F(w,q) &= w \sum_{h \in H} |\Delta u(h)|^q \\
R &= \sum_{h \in H} \frac{|\Delta z(h)|}{F(w,q)} \\
R_1(h) &= \frac{\Delta z(h)}{\Delta u(h)} \cdot \frac{1}{R} \\
R_2(h) &= \frac{\Delta u(h)}{\Delta z(h)} \cdot R \\
h^* &= \arg\max \{R_1(h_1), R_2(h_2)\}
\end{aligned}
\tag{1.23}
$$

A normalização 2 é feita com (1.24) em que $z(0)$ é o valor da função objetivo no vértice atual e z^* é o valor da solução incumbente ou um valor aspirado. A normalização só é aplicada se $z^* > z(0) - \varepsilon$, para ε pequeno. Caso $z^* \leq z(0) - \varepsilon$, ao contrário de normalizar, escolhe-se $h^* = h_2$.

$$
\begin{aligned}
R' &= \frac{z(0) - z^*}{u(0)} \\
R'_1(h) &= \frac{\Delta z(h)}{\Delta u(h)} \cdot \frac{1}{R'} \\
R'_2(h) &= \frac{\Delta u(h)}{\Delta z(h)} \cdot R' \\
h^* &= \arg\max \{R'_1(h_1), R'_2(h_2)\}
\end{aligned}
\tag{1.24}
$$

A quarta regra de escolha também usa o teste da razão, mas os movimentos em H_1 são prioritários em relação aos em H_2 colocando maior ênfase na descoberta de soluções factíveis.

Outros conceitos explorados foram a memória de longo prazo baseada em memória de freqüência, busca tabu probabilística, critério de aspiração por níveis de infactibilidade inteira e níveis de valores de função objetivo, oscilação estratégica e análise de alvo. Os resultados computacionais foram apresentados apenas para problemas de mochilas com múltiplas restrições disponíveis na OR-Library mantida por Beasley (1990).

1.3 Heurísticas baseadas em cortes de convexidade e arredondamento direcional

Glover e Laguna (1997a) utilizaram cortes de convexidade para o desenvolvimento de algoritmos heurísticos. Parte-se de $x(0)$, um vértice qualquer do poliedro definido pelas restrições do PL, quando é relaxada a integralidade das variáveis inteiras. Regiões convexas fechadas do tipo $v \leq x_j \leq v + 1$ são identificadas, com x_j sendo uma variável inteira e v o inteiro imediatamente abaixo de $x_j(0)$,

que é um valor fracionário de x_j assumido naquele vértice. Os raios extremos do cone convexo são estendidos até atingirem o hiperplano coordenado $x_j = v$ ou $x_j = v + 1$. Essa construção geométrica define um corte que exclui o vértice do PL sem excluir nenhuma solução inteira-factível. Cortes assim gerados correspondem aos cortes de Gomory (1960). Na Figura 1.3 há um exemplo no espaço reduzido das variáveis x_1 e x_2. Os raios extremos foram estendidos a partir de $x(0)$ e definem um corte ao interceptarem os hiperplanos $x_2 = 2$ e $x_2 = 1$.

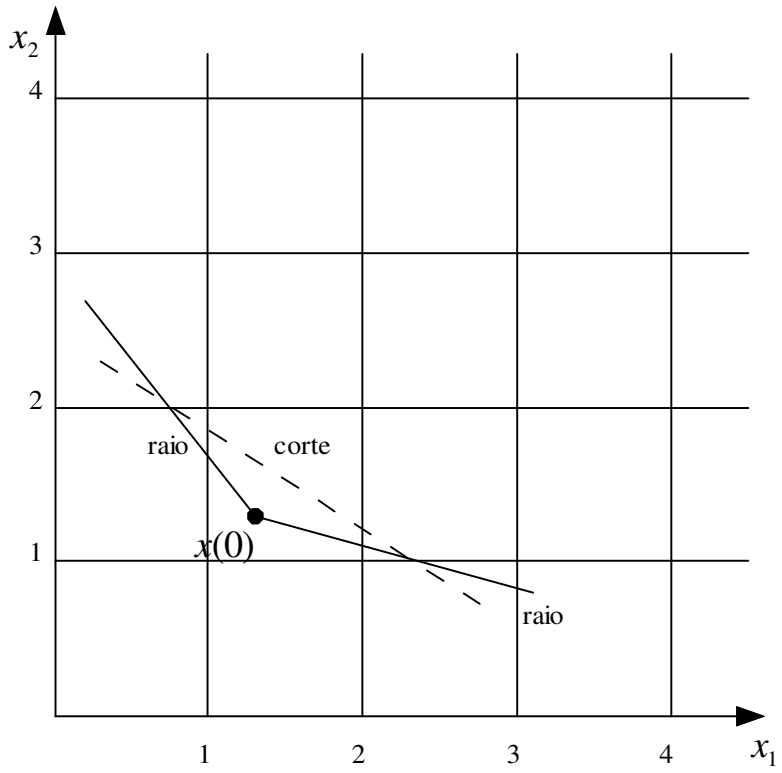

Fig. 1.3: Corte de Gomory.

A idéia central da heurística baseia-se na constatação de que os raios extremos podem ser estendidos ainda mais para interceptarem outros hiperplanos coordenados. Os hiperplanos interceptados são mantidos na forma $L'_j \leq x_j \leq U'_j$, iniciando com intervalos vazios $\lceil x_j(0) \rceil \leq x_j \leq \lfloor x_j(0) \rfloor$. Soluções inteiras potencialmente factíveis são encontradas quando todas as variáveis inteiras apresentarem intervalos $L'_j \leq x_j \leq U'_j$ não-vazios e compõem o conjunto denominado X. Essas soluções podem ser uma a uma verificadas quanto à factibilidade ou podem ser encontradas por *branch-and-cut* aplicado ao subproblema, dependendo da cardinalidade de X. Terminada a fase de extensão dos raios extremos e avaliado o conjunto X, utiliza-se um teorema que permite estender ainda mais cada raio extremo do cone convexo até atingir um hiperplano coordenado que ainda não tenha sido encontrado.

O corte condicional identificado por essas intersecções não exclui nenhuma solução inteira, exceto aquelas já presentes no conjunto X, portanto, já avaliadas.

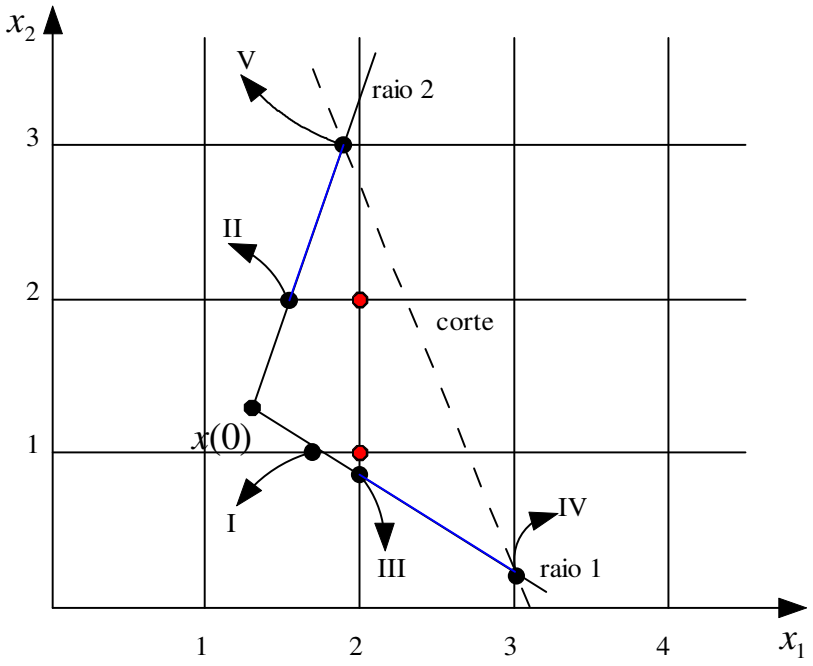

Fig. 1.4: Corte condicionado.

Por exemplo, conhecido o vértice $x(0)$ na Figura 1.4, os raios extremos 1 e 2 são estendidos, levando à seguinte seqüência de passos apresentados na Tabela 1.1.

Tab. 1.1: Exemplo de Glover e Laguna (1997a).

1. Conhecido o vértice $x(0)$	Determina-se os raios 1 e 2	$2 \leq x_1 \leq 1, 2 \leq x_2 \leq 1$
2. Estendido o raio 1	Atinge-se o ponto I	$2 \leq x_1 \leq 1, 1 \leq x_2 \leq 1$
3. Estendido o raio 2	Atinge-se o ponto II	$2 \leq x_1 \leq 1, 1 \leq x_2 \leq 2$
4. Estendido o raio 1	Atinge-se o ponto III	$2 \leq x_1 \leq 2, 1 \leq x_2 \leq 2$

As soluções inteiras $X = \{(2,1),(2,2)\}$ são verificadas quanto à factibilidade em relação às demais restrições que definem o poliedro. Pelo teorema, estendem-se os raios 1 e 2 até alcançarem um hiperplano coordenado ainda não encontrado. Os pontos IV e V são respectivamente identificados e definem o corte condicional que elimina o vértice $x(0)$ e nenhuma solução inteira, exceto as contidas em X e que já foram inspecionadas. Não há registro de implementações dessa heurística.

Combinando os cortes de convexidade com o conceito de arredondamento direcional, Glover e Laguna (1997b) apresentaram uma busca heurística conhecida como *Star-path*. Para o caso 0–1,

inicialmente proposto por Glover (1995), o arredondamento direcional $\delta(x_j^*, x_j')$ de uma variável x_j^* é definido pela expressão (1.25). Conhecido como o ponto focal do arredondamento direcional, o valor x_j' guia o arredondamento da variável x_j^*.

$$\delta(x_j^*, x_j') = \begin{cases} 0 & \text{se} \quad x_j' < x_j^* \\ 1 & \text{se} \quad x_j' > x_j^* \\ x^* & \text{se} \quad x_j' = x_j^* \text{ e } (\ x_j^* = 0 \text{ ou } x_j^* = 1) \\ 0 \text{ ou } 1 & \text{se} \quad x_j' = x_j^* \text{ e } (x_j^* \neq 0 \text{ e } x_j^* \neq 1) \end{cases} \tag{1.25}$$

A definição (1.25) é naturalmente estendida a vetores, de forma que se pode encontrar o arredondamento direcional $\delta(x^*, x')$ do vetor x^* usando x' como guia. O arredondamento para o inteiro mais próximo é um caso especial desse operador e ocorre quando cada x_j' está no intervalo entre x_j^* e o inteiro mais próximo. Se x^* estiver no interior do hipercubo unitário, qualquer ponto no segmento de reta que vai de x^* a x' é arredondado direcionalmente num mesmo vértice do hipercubo. Na Figura 1.5, por exemplo, todos os pontos da reta que passa por x^* e D^* são mapeados no vértice (1,0).

Os autores apresentam o Teorema da Imagem Convexa, que garante que a busca por soluções ótimas ou factíveis pode ser restrita à análise de pontos focais na intersecção de um hiperplano F^* com as restrições do PL. Porém, computacionalmente, é mais fácil encontrar os pontos da interseção de F^* com o cone convexo das restrições do PL com o vértice x^* em algum ponto extremo do poliedro. Uma maneira conveniente é usar o ponto extremo ótimo do PL como sendo x^* e encontrar um corte de convexidade F^* que o exclua sem excluir nenhuma solução inteira-factível. Por exemplo, na Figura 1.5, o corte F^* passa por A^* e D^*, embora a região factível esteja compreendida entre A e D.

Cada região convexa da intersecção de F^* com a região factível do PL pode ser biunivocamente mapeada, por arredondamento direcional, em um vértice factível do hipercubo unitário. Assim, é proposto um mecanismo eficiente de gerar trajetórias P (segmento de reta) a partir de combinações convexas entre pares de pontos em F^*. Os pontos dos raios extremos do cone com vértice em x^*, e que permitiram derivar o corte F^*, são os utilizados para gerar as trajetórias P. Tomando novamente a Figura 1.5 como exemplo, se o primeiro ponto escolhido for o A^*, o segundo ponto obrigatoriamente será o D^*. Note que quando há apenas duas dimensões, só há uma possibilidade de trajetória P, já que o hiperplano em questão também é uma reta. Ao percorrer essa trajetória linear entre esses pontos, cada trecho convexo em P é mapeado, por arredondamento direcional, em um único vértice inteiro do hipercubo unitário. Os pontos de A a B são mapeados em (0,1), os pontos de B a C são mapeados em (1,1) e os de C a D em (1,0). Note que o ponto B pode ser mapeado em (0,1) ou (1,1), enquanto que C pode ser em (1,1) ou (1,0), pois ambos estão sobre as linhas pontilhadas que são tratadas pelo pelo caso especial da expressão (1.25), $x_j' = x_j^*$ e ($x_j^* \neq 0$ e $x_j^* \neq 1$). Enquanto o caminho P é uma trajetória contínua contida em F^*, o arredondamento direcional dos pontos de P,

$\delta(x^*, P)$ é descontínuo e forma o *Star-path.*

É importante que essas trajetórias evitem passar novamente por regiões de F^* cujos arredondamentos direcionais levem a vértices já identificados por outras trajetórias percorridas. Para atender essa necessidade, os autores propõem um procedimento que gera trajetórias P que procuram manter um espalhamento ao longo do hiperplano F^*. Em cada trajetória P, um dos pontos usados jaz sobre um dos raios extremos do cone convexo que foi utilizado para construir o corte F^*. O outro ponto encontrado é o centro de gravidade dos demais pontos que definem F^*. O trabalho também propõe a inclusão do *Star-path* em mecanismos baseados em *scatter search* (Glover et al., 2000), e generaliza o método para variáveis inteiras gerais. Não há registro de implementações dessa heurística.

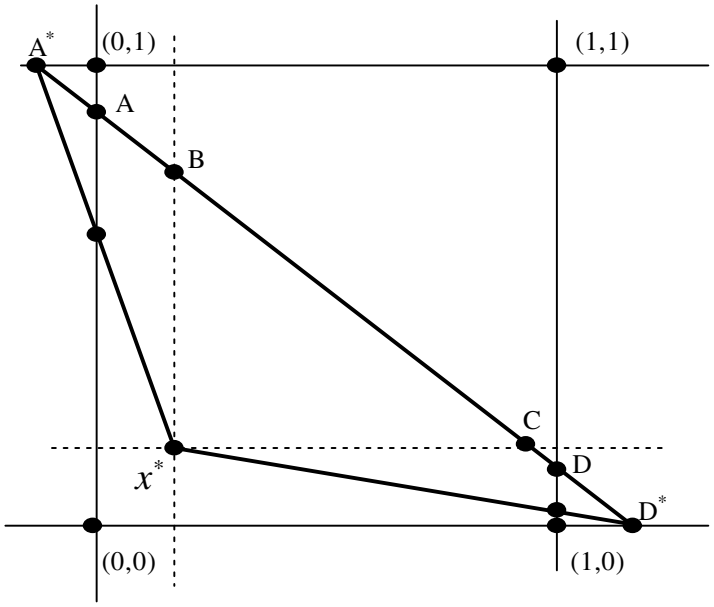

Fig. 1.5: Arredondamento direcional.

Balas et al. (2001) desenvolveram a heurística Octane (*Octahedral Neighborhood Enumeration*) para minimização em problemas de programação inteira 0–1 pura. Dado o cubo n-dimensional K centrado na origem (1.26), o algoritmo usa um octaedro n-dimensional K^* a ele circunscrito que é determinado por (1.27). Todas as componentes do vetor e são tais que $e_j = 1\ \forall j \in \{1, \ldots, n\}$. As componentes do vetor δ são tais que $\delta_j \in \{-1, 1\}\ \forall j \in \{1, \ldots, n\}$. Na Figura 1.6 há um cubo para $n = 2$, enquanto que a Figura 1.7 complementa o exemplo com o octaedro circunscrito para duas dimensões.

$$K = \left\{ x \in R^n \,\middle|\, \frac{-e}{2} \leq x \leq \frac{e}{2} \right\} \tag{1.26}$$

$$K^* = \left\{ x \in R^n \,\middle|\, \delta x \leq \frac{n}{2}, \forall \delta \in \{\pm 1\}^n \right\} \tag{1.27}$$

Há um mapeamento 1 para 1 entre cada faceta do octaedro e cada um dos 2^n vértices do cubo, pois a faceta $\delta x = \frac{n}{2}$ contém um e somente um vértice v de K, de forma que $v_j = \frac{1}{2}$ se $\delta_j = 1$ e $v_j = -\frac{1}{2}$ se $\delta_j = -1$. Os pontos do problema original são transladados para o modelo com centro na origem pela expressão $x' = x - \frac{e}{2}$.

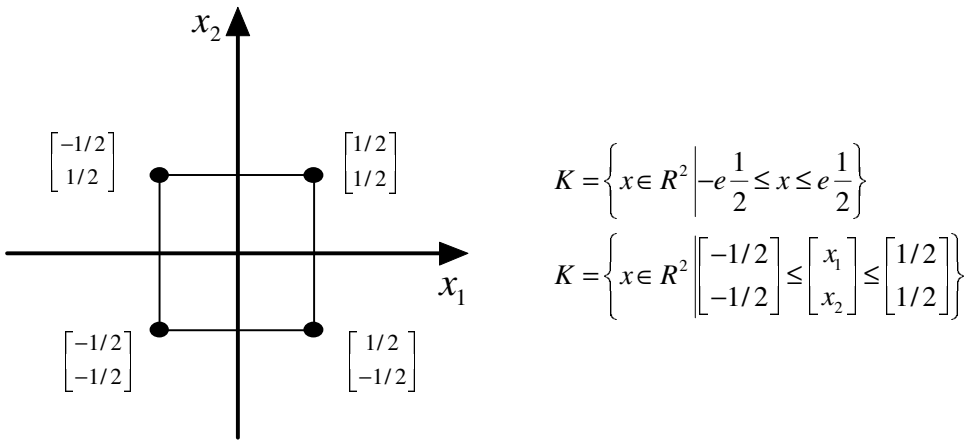

Fig. 1.6: Cubo em duas dimensões.

A heurística parte de um ponto PL-factível, $\bar{x}$, normalmente a solução ótima do PL, e segue um raio segundo uma direção D escolhida, conforme a expressão (1.28) até interceptar as primeiras k facetas do octaedro K^*. Este procedimento encontra k vértices do cubo K, e cada um deles é testado em relação à factibilidade do PL. No exemplo da Figura 1.8 estão rotulados os dois pontos de intersecção I e II e os respectivos vértices I' e II' do cubo.

$$x = \bar{x} + \lambda D, \ \lambda > 0 \tag{1.28}$$

A determinação da direção D é um ponto crucial na heurística. Quatro métodos foram propostos, levando em consideração que a heurística será usada em conjunto com um algoritmo de *branch-and-cut*.

- *Média dos raios*: a média dos raios extremos normalizados do cone convexo formado pela base ótima do PL.
- *Raio objetivo*: caminhar na direção oposta da função objetivo, $-c$, para dentro do poliedro.
- *Raio diferença*: usar o vetor diferença entre a solução ótima do nó corrente do *branch-and-cut* e do nó raiz.

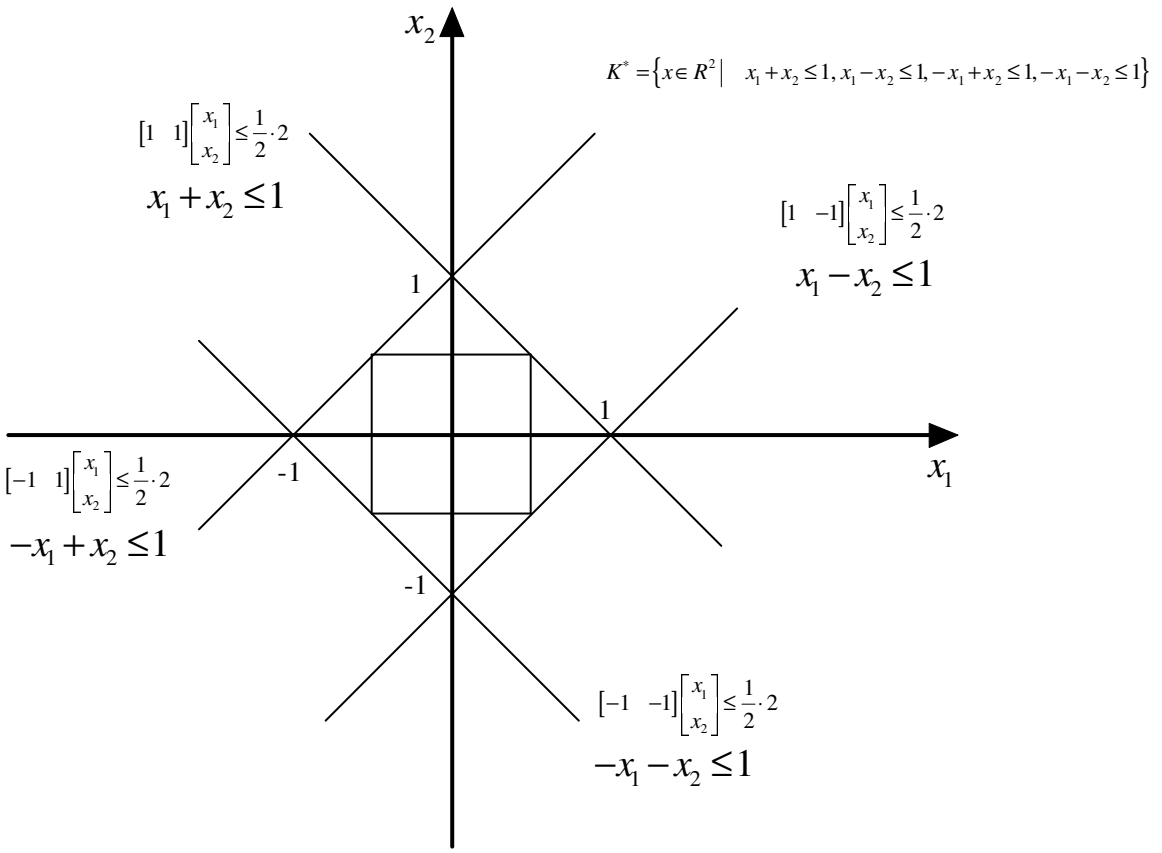

Fig. 1.7: Octaedro circunscrito ao cubo para 2 dimensões.

- *Média ponderada*: usar os raios extremos como na média dos raios, atribuindo como peso o inverso do seu custo reduzido, mas usar apenas aqueles cujas variáveis não-básicas associadas são de folga e apresentem custo reduzido positivo.

Há um algoritmo que de forma eficiente enumera os k primeiros hiperplanos interceptados. A heurística foi aplicada a 21 instâncias compreendidas entre problemas da MIPLIB 3, 3 problemas GAP da OR-Library mantida por Beasley (1990) e um problema de caixeiro viajante. O número de variáveis binárias nas instâncias varia de 27 a 2756. Os resultados computacionais foram competitivos com Balas et al. (2004).

Glover (2007) propôs heurísticas para PI baseadas em arredondamento direcional de soluções fracionárias selecionadas em raios do cone convexo associado ao vértice da solução ótima da relaxação PL do PIM. A abordagem permite trajetórias factíveis e infactíveis relativas à fronteira da relaxação linear; sua motivação é o apelo geométrico e o sucesso de abordagens similares em heurísticas para resolver problemas combinatórios. Cada iteração do processo envolve o arredondamento de uma variável selecionada, que é induzido por meio de inequações de ramificação que são adicionadas ao PL.

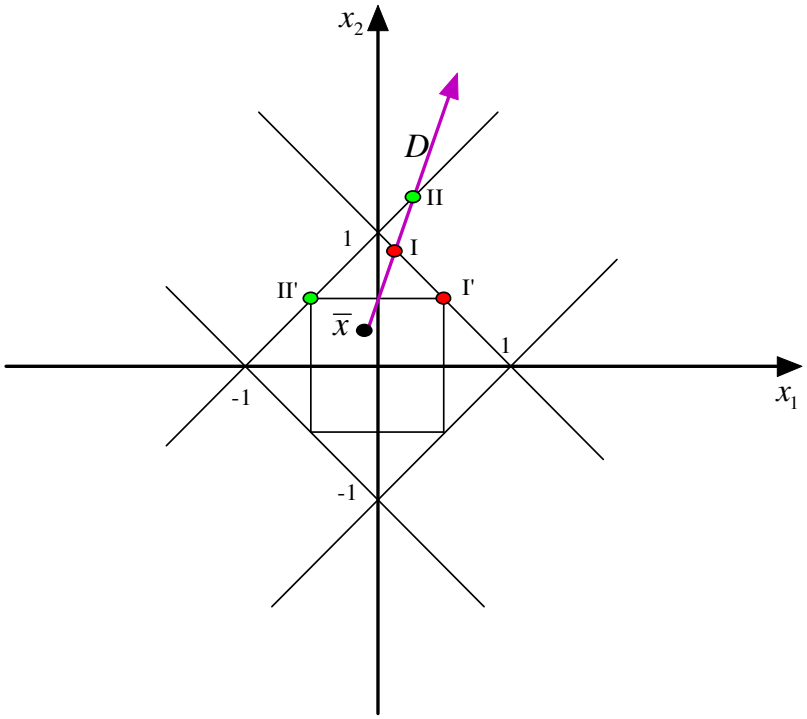

Fig. 1.8: Exemplo da heurística Octane.

A solução ótima do novo PL, obtida por reotimização do PL corrente, mostra as implicações do arredondamento naquela variável. O vértice ótimo do novo PL gera um novo cone, novos raios e novas soluções direcionalmente arredondadas. Não foram apresentados resultados computacionais.

1.4 Heurísticas de busca local com uso de pacotes comerciais

O sucesso apresentado pelos pacotes comerciais como Cplex (2009) e Xpress (2009) a partir do final dos anos 90 motivou o surgimento de heurísticas de busca local que são aplicadas tão logo a primeira solução inteira-factível tenha sido encontrada pelo resolvedor. Fischetti e Lodi (2003) propuseram a ramificação local (*Local Branching*), que é um procedimento que usa busca local na vizinhança da solução incumbente. Desigualdades inválidas são introduzidas num nível estratégico para definir a vizinhança em que a busca local será aplicada, enquanto que as decisões táticas são delegadas ao pacote que explora a vizinhança. O objetivo dessa abordagem de dois níveis é atualizar o máximo possível a solução incumbente nos estágios iniciais da busca do *branch-and-cut*.

A ramificação local considera as variáveis binárias $x_j \in \{0,1\}\,,\ \forall j \in \mathcal{B}$, uma solução de referência $\bar{x}$ que é inteira-factível e um parâmetro inteiro positivo k. A vizinhança k-*OPT* de $\bar{x}$, $\mathcal{N}(\bar{x},k)$, é o conjunto de soluções inteiras-factíveis que satisfazem a desigualdade (1.29) em que

$\mathcal{B}_1 = \{j \in \mathcal{B} \mid \bar{x}_j = 1\}$.

$$\Delta(x, \bar{x}) = \sum_{j \in \mathcal{B}_1} (1 - x_j) + \sum_{j \in \mathcal{B} \setminus \mathcal{B}_1} x_j \leq k \tag{1.29}$$

Os dois termos à esquerda da desigualdade correspondem ao número de variáveis binárias que mudam de 1 para 0 e de 0 para 1, respectivamente. O nó de ramificação corrente é particionado pela disjunção (1.30).

$$\Delta(x, \bar{x}) \leq k \text{ (ramo à esquerda)} \quad \text{ou} \quad \Delta(x, \bar{x}) \geq k + 1 \text{ (ramo à direita)} \tag{1.30}$$

É desejável que a vizinhança $\mathcal{N}(\bar{x}, k)$ correspondente ao ramo à esquerda seja suficientemente pequena para ser otimizada em tempo computacional pequeno, mas grande o suficiente para conter soluções melhores que $\bar{x}$. Na Figura 1.9, $\bar{x}^1$ é a solução de partida no nó 1 (solução incumbente). Cada triângulo com a letra T corresponde a uma subárvore explorada pelo Cplex no nível tático. A solução incumbente é atualizada quando $\bar{x}^2$, melhor que $\bar{x}^1$ e ótima na vizinhança $\mathcal{N}(\bar{x}^1, k)$, é encontrada no nó 2. O procedimento é então reaplicado ao nó 3, em que a exploração da vizinhança $\mathcal{N}(\bar{x}^2, k) \setminus \mathcal{N}(\bar{x}^1, k)$ produz uma nova solução incumbente $\bar{x}^3$ no nó 4. O nó 5 corresponde ao problema original PIM com as restrições de ramificação $\Delta(x, \bar{x}^1) \geq k + 1$ e $\Delta(x, \bar{x}^2) \geq k + 1$. A exploração do nó 6 não produz uma solução melhorada e a restrição de ramificação $\Delta(x, \bar{x}^3) \geq k + 1$ leva ao nó 7. A solução ótima nos ramos à esquerda é obtida pelo pacote de otimização Cplex.

A ramificação local assim definida tem natureza exata, mas pode ser usada de maneira heurística pela imposição de um tempo limite (parâmetro) na exploração de cada ramo à esquerda, tempo esse que quando excedido conduz a dois casos:

a) Se a solução incumbente foi melhorada, volta-se ao nó pai e cria-se outro ramo à esquerda com o mesmo valor de k usando a nova solução como referência.

b) Se a solução incumbente não foi melhorada, volta-se ao nó pai e cria-se outro ramo à esquerda em que k é reduzido, por exemplo para $\lceil k/2 \rceil$, isto é, reduz-se a vizinhança para acelerar a exploração.

A diversificação na busca do espaço de soluções é conduzida no espírito de busca em vizinhança variável (*Variable Neighborhood Search*) proposta por Mladenović e Hansen (1997), ao se aumentar a vizinhança corrente em, por exemplo $\lceil k/2 \rceil$. O número de vezes que a diversificação é acionada é um parâmetro e o critério de parada do método heurístico de ramificação local é o tempo computacional.

Nos experimentos computacionais, a ramificação local produziu soluções de melhor qualidade que o Cplex 7.0 com tempo computacional limitado a 1, 3 e 5 horas. Para 29 instâncias coletadas da MIPLIB 3 e de outros autores a ramificação local apresentou resultados melhores que o Cplex 7.0 em 23 instâncias.

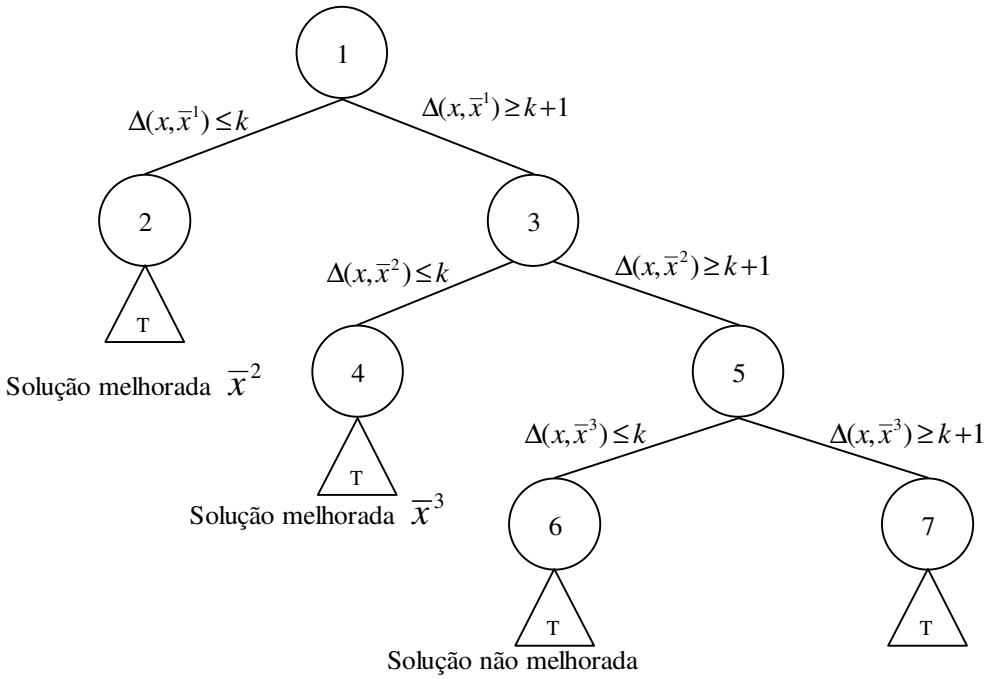

Fig. 1.9: Ramificação local.

Hansen et al. (2006) adaptou o método de ramificação local em um procedimento de *Variable Neighborhood Search* com um controle mais sistemático na vizinhança. Na fase *shaking*, k_{shak} inicia em 1 e é incrementado em uma unidade a cada passo. Na fase busca em descida com vizinhança variável (*Variable Neighborhood Descent*), o parâmetro k é incrementado com k_{step} a cada passo e inicia com o próprio k_{step}. O limite superior tanto para k_{shak} quanto para k é n. Experimentos computacionais mostram que a modificação é efetiva em relação à ramificação local, com resultados melhores para 19 casos dentre as mesmas 29 instâncias utilizadas em Fischetti e Lodi (2003).

A ramificação local usa a fixação leve de variáveis, pois nenhuma variável é obrigatoriamente mantida num valor fixo, permitindo, por outro lado, alterar um número limitado k de variáveis. Danna et al. (2005) criaram uma outra heurística de busca local que usa a fixação forte de variáveis, uma vez que variáveis são fixadas em valores específicos. Esta heurística, denominada *Relaxation Induced Neighborhood Search* (RINS), também faz uso intensivo de resolvedores e utiliza a informação da solução incumbente e da relaxação linear do PIM em um nó qualquer da árvore do *branch-and-cut*. A solução incumbente serve como guia e atende o requisito da factibilidade, já a solução do PL num nó do *branch-and-cut* tem valor de função objetivo melhor que o da solução incumbente, embora falhe em atender a factibilidade inteira. Essas observações conduziram ao procedimento de três passos.

1. Fixar as variáveis que assumem os mesmos valores na solução incumbente e na solução da

relaxação linear do PIM.

2. Estabelecer uma restrição de qualidade da função objetivo baseada no valor da solução incumbente.

3. Resolver o sub-PIM nas variáveis restantes que não foram fixadas.

A heurística é aplicada com freqüência $f \gg 1$ nos nós do *branch-and-cut*, e limita o número de nós inspecionados no sub-PIM em nl. A aplicação da heurística em nós relativamente distintos promove a diversificação no método. Para as 37 instâncias de teste utilizadas e coletadas da MIPLIB e outras fontes, apresentou resultados superiores à ramificação local.

Ghosh (2007) desenvolveu uma heurística híbrida[2] que combina a ramificação local com RINS. A heurística denominada *Distance Induced Neighborhood Search* (DINS) vale-se do arrazoado de que soluções inteiras-factíveis estão provavelmente mais próximas da relaxação linear do nó corrente da árvore de enumeração. Seja $x_{j(pim)}$ o valor assumido por x_j na solução incumbente e $x_{j(nó)}$ o valor de x_j na relaxação linear do nó corrente. Como a restrição que pode controlar a distância euclideana $\sum_{j \in \mathcal{N} \cup \mathcal{C}} \left(x_j - x_{j(nó)}\right)^2 \leq \sum_{j \in \mathcal{N} \cup \mathcal{C}} \left(x_{j(pim)} - x_{j(nó)}\right)^2$ não é linear, ela é substituída por redefinições nos limitantes superiores e inferiores das variáveis inteiras canalizadas. Essas redefinições são inspiradas na desigualdade $\sum_{j \in \mathcal{B} \cup \mathcal{G}} \left|x_j - x_{j(nó)}\right| \leq \sum_{j \in \mathcal{B} \cup \mathcal{G}} \left|x_{j(pim)} - x_{j(nó)}\right|$. Seja a variável inteira geral x_j com $\left|x_{j(pim)} - x_{j(nó)}\right| < 0,5$. Se x_j assumir valor diferente de $x_{j(pim)}$ numa nova solução incumbente, essa distância aumentará. Então, para que não ocorra aumento da distância sob essa condição, a variável inteira geral é fixada em $x_{j(pim)}$. Por outro lado, se a variável inteira geral x_j apresentar $\left|x_{j(pim)} - x_{j(nó)}\right| \geq 0,5$ e assumir valor diferente de $x_{j(pim)}$ numa nova solução incumbente, a distância não necessariamente aumentará, então para garantir que não haja acréscimo na distância, os limites l_j e u_j são redefinidos como em (1.31):

$$\begin{array}{l} \text{se } x_{j(pim)} > x_{j(nó)} \text{ então } \left\{ \begin{array}{rcl} l_j^{novo} & = & \max\left(l_j^{velho}, \left(\left\lceil x_{j(nó)} - \left(x_{j(pim)} - x_{j(nó)}\right)\right\rceil\right)\right) \\ u_j^{novo} & = & x_{j(pim)} \end{array} \right. \\ \text{se } x_{j(pim)} < x_{j(nó)} \text{ então } \left\{ \begin{array}{rcl} l_j^{novo} & = & x_{j(pim)} \\ u_j^{novo} & = & \min\left(u_j^{velho}, \left(\left\lfloor x_{j(nó)} + \left(x_{j(nó)} - x_{j(pim)}\right)\right\rfloor\right)\right) \end{array} \right. \end{array} \tag{1.31}$$

Na Figura 1.10 há um exemplo dos ajustes nos limitantes de uma variável x_j. Antes dos ajustes, $l_j^{velho} = 0$ e $u_j^{velho} = 10$. No nó corrente $x_{j(nó)} = 5{,}4$ e $x_{j(pim)} = 8$. Como $x_{j(pim)} > x_{j(nó)}$,

[2]Neste capítulo, heurística híbrida é usada no sentido de combinar duas ou mais idéias elementares que compõem o fundamento de heurísticas distintas.

$u_j^{novo} = x_{j(pim)}$. A distância de $x_{j(nó)}$ ao seu limitante superior u_j^{novo} passa a ser $x_{j(pim)} - x_{j(nó)} = 2{,}6$. Essa distância é então utilizada para ajustar o limitante inferior, $l_j^{novo} = \max\left(0, \left(\lceil x_{j(nó)} - 2{,}6)\rceil\right)\right)$ $= \max\left(0, (\lceil 5{,}4 - 2{,}6)\rceil\right)) = 3$.

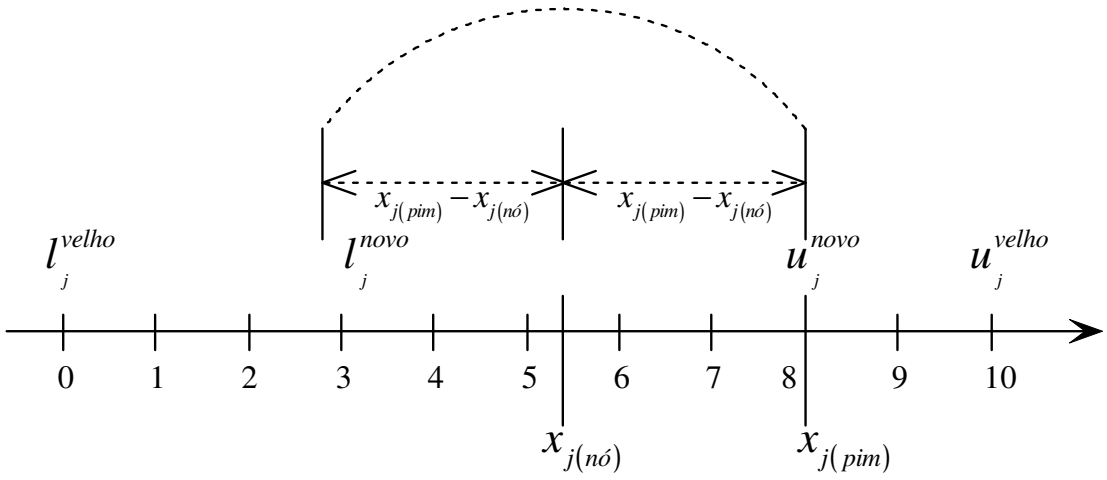

Fig. 1.10: Exemplo de ajustes dos limitantes de uma variável na heurística DINS.

Variáveis 0–1 não sofrem ajustes em $l_j = 0$ e $u_j = 1$ pela expressão (1.31). Aquelas que apresentam $\left|x_{j(pim)} - x_{j(nó)}\right| < 0,5$ são fixadas em $x_{j(pim)}$. As demais com $\left|x_{j(pim)} - x_{j(nó)}\right| \geq 0,5$ são fixadas em $x_{j(pim)}$ se $x_{j(pim)} = x_{j(nó)} = x_{j(raiz)}$ e $\Delta_j = \textit{falso}$. Cada componente Δ_j do vetor Δ assume o valor *falso*, se x_j recebeu o mesmo valor em todas as soluções incumbentes encontradas até aquele momento, e *verdadeiro*, caso contrário. Os valores $x_{j(raiz)}$ são os assumidos pelas variáveis na relaxação linear do PIM na raiz da árvore do *branch-and-cut*. As variáveis 0–1 que não foram fixadas são incluídas em uma restrição de ramificação local dada por

$$\sum_{j \in \mathcal{B}-F \text{ e } x_{j(\text{pim})}=1} (1 - x_j) \quad + \sum_{j \in \mathcal{B}-F \text{ e } x_{j(\text{pim})}=0} x_j \quad \leq p \tag{1.32}$$

Na desigualdade (1.32), $\mathcal{B}$ é o conjunto das variáveis 0–1 enquanto que F é o conjunto das variáveis 0–1 que foram fixadas. O sub-PIM assim definido é resolvido com limite de nl nós. Caso o limite nl seja atingido, p é reduzido iterativamente em 5 unidades e o sub-PIM é novamente resolvido enquanto p não se tornar negativo. A heurística DINS é reaplicada a cada f nós do *branch-and-cut* do problema original e é aplicada pela primeira vez quando a primeira solução inteira-factível é encontrada. Os resultados computacionais conduzidos com as mesmas instâncias de Danna et al. (2005) superaram tanto a ramificação local quanto a RINS.

Santos (2006) mostra que as desigualdades de ramificação utilizadas na heurística ramificação local são casos particulares dos cortes canônicos introduzidos por Balas e Jeroslow (1972). Baseado

nessa observação, é proposta uma variante da ramificação local com a incorporação de cortes canônicos mais profundos, ou seja, aqueles que descartam mais soluções. Foram realizados testes com 25 das 29 instâncias utilizadas por Fischetti e Lodi (2003) com a obtenção de resultados de qualidade superior aos alcançados pela ramificação local. Também foram observados melhores resultados em relação ao *branch-and-cut* do Xpress versão 16.10.3. O autor também combinou os cortes canônicos com a heurística RINS, alcançando resultados promissores para o mesmo conjunto de instâncias.

1.5 A heurística *Feasibility Pump* e suas variações

Fischetti et al. (2005) propuseram uma heurística chamada *Feasibility Pump* (FP) que pela reotimização de sucessivos programas lineares tenta encontrar uma solução inteira-factível para minimização em problemas PIM 0–1. A heurística FP gera duas trajetórias dos pontos $\tilde{x}$ e x^* que em geral convergem. De um lado, $\tilde{x}$ satisfaz apenas a integralidade inteira e do outro lado x^* satisfaz apenas as restrições lineares, ou seja, $x^* \in P$, em que P é o poliedro definido pelo conjunto das restrições do PL.

O método inicia com um ponto x^*, a solução ótima do PL. O ponto $\tilde{x}$, tipicamente infactível, é encontrado ao arredondar x^* para o inteiro mais próximo conforme a expressão (1.33).

$$\tilde{x}_j = \begin{cases} [x_j] & \text{se} \quad j \in \mathcal{B} \\ x_j & \text{c.c.} \end{cases} \tag{1.33}$$

$$\Delta(x, \tilde{x}) = \sum_{j \in \mathcal{B}} |x_j - \tilde{x}_j| = \sum_{j \in \mathcal{B} | \tilde{x}_j = 0} x_j + \sum_{j \in \mathcal{B} | \tilde{x}_j = 1} (1 - x_j) \tag{1.34}$$

Na expressão (1.33), $[.]$ representa o arredondamento escalar para o inteiro mais próximo e $\mathcal{B}$ é o conjunto das variáveis binárias. A distância entre um ponto $x \in P$ e um ponto inteiro $\tilde{x}$ é definida pela norma L_1 da expressão (1.34). Dado um ponto inteiro $\tilde{x}$, o ponto x^* mais próximo dele e que pertence ao poliedro P pode ser determinado ao resolver o PL (1.35).

$$\min\{\Delta(x, \tilde{x}) | Ax + Dy \geq b\} \tag{1.35}$$

A minimização da distância $\Delta(x, \tilde{x})$ desencoraja a solução x^* do PL a se afastar demais de $\tilde{x}$ à medida que o método progride gerando novos pares x^* e $\tilde{x}$ conforme o algoritmo da Figura 1.11.

No algoritmo, nIT mantém o total de iterações transcorridas e o ponto inicial x^* é dado pela solução do PL original no *Passo* 1. Após um tempo máximo *TL* haver transcorrido sem sucesso em obter uma solução, a heurística termina com falha. No *Passo* 9, *T* é um parâmetro da heurística que controla o número de variáveis *TT* que são complementadas. *TT* é uma perturbação aleatória sorteada

com distribuição uniforme no intervalo $\left[\frac{T}{2}, \frac{3T}{2}\right]$ para evitar ciclagem, problema inerente da heurística. Esse mecanismo é acionado quando no *Passo* 8 o arredondamento da solução ótima do PL, x^*, é igual à última solução arredondada $\tilde{x}$. A heurística foi testada em 44 instâncias da Miplib 2003 e outras 39 instâncias não disponíveis abertamente. Falhou em 3 instâncias enquanto que a heurística de nó do Cplex 8.1 falhou em 19. A FP obteve melhor valor de função objetivo em 46 casos e perdeu em 33. Bertacco et al. (2007) estenderam a heurística FP para PIM com variáveis inteiras gerais e obtiveram soluções factíveis em instâncias da Miplib 2003, Mittelmann (2003) e Danna et al. (2005) em tempo reduzido.

Fischetti e Lodi (2008) ao trabalharem com $\mathcal{B} \neq \emptyset$ em PIM geral, combinaram o procedimento da ramificação local com o *Feasibility Pump*. Como a ramificação local exige uma solução de referência factível inteira, o *Feasibility Pump* é aplicado por um tempo computacional muito pequeno (no máximo alguns segundos) e encontra uma solução próxima da factibilidade. Embora a solução encontrada $\hat{x}$ seja inteira, algumas restrições são violadas e passam a compor o conjunto T. O procedimento da ramificação local é então aplicado em um problema PIM modificado. Para cada $i \in T$, a restrição original $a_i x \geq b_i$ é relaxada em $a_i x + \sigma_i \geq b_i$ em que $\sigma_i \geq 0$ é uma variável contínua e artificial. Além disso, uma nova restrição é criada para cada σ_i, na forma

$$\sigma_i \leq \delta_i y_i, \quad y_i \in \{0, 1\} \tag{1.36}$$

Na desigualdade (1.36), δ_i é um valor grande, enquanto que y_i é uma nova variável binária. Note que em (1.36), se $y_i = 0$, então $\sigma_i = 0$. Se nesse problema relaxado for possível encontrar uma solução inteira-factível em que todo $y_i = 0$, essa solução também será factível para o PIM original. Portanto a função objetivo é alterada de $\min cx$ para $\min \sum_{i \in T} y_i$. Esse modelo relaxado se encaixa bem na abordagem da ramificação local que é aplicada nas $|\mathcal{B}| + |T|$ variáveis binárias. Como a função objetivo alterada não sofre nenhuma influência da função objetivo do problema original, em geral a solução encontrada é de baixa qualidade. Porém essa solução pode servir de referência para uma nova etapa de ramificação local aplicada agora ao problema original.

Em um universo de 83 instâncias da Miplib 2003 e outras pertencentes aos autores, alcançou uma primeira solução factível em menor tempo que o Cplex 9.0.3 em 44 casos.

A heurística FP tem o grande trunfo de encontrar soluções factíveis em tempo computacional pequeno, muitas vezes na ordem de segundos. Porém, nem sempre a solução encontrada tem qualidade satisfatória. Para contornar essa desvantagem, Achterberg e Berthold (2007) propuseram uma pequena variação na heurística. A função objetivo passa a ser composta de uma combinação convexa de $\Delta(x, \tilde{x})$ com a função objetivo original do PL, $cx + dy$. Inicialmente a ênfase está integralmente em $cx + dy$, mas com o passar das iterações, geometricamente se desloca para $\Delta(x, \tilde{x})$. Aplicada a instâncias da Miplib 2003, Mittelmann (2003) e Danna et al. (2005), apresentou resultados promissores,

Passo 1	$nIT \leftarrow 0$, e $x^* = \arg\min\{cx + dy \mid Ax + Dy \geq b\}$
Passo 2	Se x^* for inteiro, então vá ao *Passo* 10
Passo 3	$\tilde{x} \leftarrow [x^*]$
Passo 4	Se tempo $>$ *TL*, então vá ao *Passo* 10
Passo 5	$nIT \leftarrow nIT + 1$
Passo 6	$x^* = \arg\min\{\Delta(x, \tilde{x}) \mid Ax + Dy \geq b\}$
Passo 7	Se x^* for inteiro, então vá ao *Passo* 10
Passo 8	Se $\exists\ j \in \mathcal{B} \mid [x^*] \neq \tilde{x}_j$ então $\tilde{x}_j \leftarrow [x^*]$ e vá ao *Passo* 4
Passo 9	Complemente $TT = rand(\frac{T}{2}, \frac{3T}{2})$ variáveis $\tilde{x}_j (j \in \mathcal{B})$ com maiores valores de $\left\|x_j^* - \tilde{x}_j\right\|$ e vá ao *Passo* 4.
Passo 10	FIM

Fig. 1.11: Heurística *Feasibility Pump*.

embora o número de instâncias sem solução encontrada tenha aumentado em relação à FP.

A Tabela 1.2 resume as idéias centrais usadas nas heurísticas apresentadas. Tanto a fixação fraca (ramificação local) quanto a fixação forte, sempre vêm acompanhadas do uso de resolvedores aplicados ao sub-PIM. O uso de direção por raio e pivotamentos é predominante na maioria das heurísticas. Note também que o uso de memória tabu não foi muito explorado na literatura. A coluna PL-modificado é relativa a qualquer idéia que resolva alguma variação do PL do problema original. Por exemplo, Ibaraki et al. (1974) e Faaland e Hillier (1979) utilizam um PL parametrizado e que cria um caminho linear por partes no interior do poliedro. Já a heurística *Feasibility Pump* e suas variações usam a minimização da distância da solução do PL em relação à solução arredondada. Na coluna "busca local", são registradas apenas as heurísticas que constroem internamente um mecanismo de vizinhança. Heurísticas que usam sub–PIM em vizinhança induzida como a ramificação local não recebem esse rótulo. Na coluna "cortes", são também registradas as heurísticas que usam cortes inválidos como a ramificação local.

Tab. 1.2: Fundamentos usados nas heurísticas.

Heurística	Busca local	Fix. fraca	Fix. forte	Direção por raio	Sub PIM	PL-modificado	Mem. tabu	Pivota-mentos	Cortes
Hillier (1969)	■			■					
Ibaraki et al. (1974)	■			■		■			
Faaland e Hillier (1979)	■			■		■			
Saltzman e Hillier (1992)	■			■					
Balas e Martin (1980)	■							■	
Balas et al. (2004)		■	■		■			■	
Løkketangen et al. (1994)	■						■	■	
Eckstein e Nediak (2007).			■		■			■	■
Løkketangen e Glover (1998)							■	■	
Glover e Laguna (1997a)					■				■
Glover e Laguna (1997b)				■					■
Glover (2007)			■	■					
Balas et al. (2001)				■					■
Fischetti e Lodi (2003)		■			■				■
Hansen et al. (2006)		■			■				■
Fischetti e Lodi (2008)		■		■	■	■			■
Danna et al. (2005)			■		■				
Ghosh (2007)		■	■		■				■
Santos (2006)		■	■		■				■
Fischetti et al. (2005)				■		■			
Bertacco et al. (2007)				■		■			
Achterberg e Berthold (2007)				■		■			

Capítulo 2

Busca Tabu Paramétrica

Neste capítulo os fundamentos da busca tabu paramétrica são abordados, culminando com um algoritmo que faz uso de memória tabu de curto prazo. Inicialmente, apresentamos uma breve discussão sobre as limitações da busca em árvore, além de dois métodos predecessores da busca tabu paramétrica.

2.1 A inflexibilidade da memória estática da busca em árvore

Em implementações de *branch-and-bound,* durante a construção da árvore de enumeração, as decisões de ramificação tomadas são irreversíveis e os ramos descendentes herdam as limitações impostas pelos seus predecessores. As decisões do estágio inicial da enumeração exercem forte influência no desempenho dos algoritmos de *branch-and-bound*. Por exemplo, a descoberta precoce de uma solução de alta qualidade pode eliminar várias subárvores ainda não exploradas que apresentam limitantes de pior qualidade do que dessa solução. Idealmente, as primeiras ramificações impostas devem ter o efeito de limitar o máximo possível o tamanho da árvore. Porém, no início da busca, as decisões tomadas são baseadas em informações muito limitadas com respeito à expectativa de que um ramo escolhido possa levar à descoberta de uma solução de alta qualidade. Os métodos de enumeração baseados na relaxação linear do problema efetuam a ramificação na variável que maximiza a melhoria no limitante inferior (Atamtürk e Savelsbergh, 2005). O cálculo de penalidades UP/DN ($x_j \geq \lceil \bar{x}_j \rceil$ e $x_j \leq \lfloor \bar{x}_j \rfloor$ em que $\bar{x}_j$ é um valor fracionário assumido na solução ótima do PL num dado nó da árvore para a variável x_j) de Tomlin (1971) aplica um passo dual simplex implicitamente em cada variável fracionária. A sua extensão natural, conhecida como ramificação forte total (*full strong branching*), reotimiza o PL até a otimalidade e é computacionalmente proibitiva se aplicada a todos os nós filhos associados com as ramificações. Bénichou et al. (1971) observaram que após sucessivas ramificações em uma mesma variável, a média da degradação na função objetivo estabiliza

para essa variável. Baseado nessa observação, propuseram uma medida de degradação conhecida por pseudocusto. Nessa abordagem, variáveis cujos pseudocustos não foram inicializados recebem tratamento de *strong branching* para inicializá-los. Achterberg et al. (2005) generalizaram o pseudocusto num método nomeado *Reliability Branching* de forma que variáveis que ainda não têm pseudocustos confiáveis recebem tratamento de *strong branching*.

Essas técnicas de escolha de ramificação podem falhar. Às vezes, em estágios iniciais da árvore, os ramos selecionados podem ser pouco significativos a ponto de influenciarem de forma efetiva as próximas ramificações. Referências relevantes que versam sobre esses temas são encontradas em Linderoth e Savelsbergh (1999), Johnson et al. (2000) e Atamtürk e Savelsbergh (2005).

A abordagem do *branch-and-bound* dinâmico (Glover e Tangedahl, 1976) e (Hanafi e Glover, 2002) suscita alternativas estratégicas não consideradas nos algoritmos tradicionais de *branch-and-bound*. Nesse contexto, a idéia é reavaliar e eliminar ramos sem influência quando uma informação mais confiável surge em estágios posteriores da busca. Para tornar mais claros esses conceitos, um exemplo é apresentado na Figura 2.1 para o seguinte problema de programação inteira extraído de Glover e Tangedahl (1976) no qual todas as variáveis são inteiras.

$$
\begin{array}{l}
\min 8x_1 + 8x_2 - 4x_3 \\
\text{sujeito a} \\
\begin{array}{rrrcr}
3x_1 & & -2x_3 & \geq & 2 \\
2x_1 & -2x_2 & & \geq & 1 \\
-8x_1 & +20x_2 & & \geq & -1
\end{array} \\
x_j \geq 0 \text{ e inteira, } j \in \{1,2,3\}
\end{array}
$$

No passo 2 da Figura 2.1A, conclui-se que a imposição $x_1 \geq 1$ que ocorreu no Passo 1, não está mais ativa pois após a adição do segundo ramo no Passo 2, observa-se que $x_1 = 1,33$. Esse ramo é classificado como condicionalmente supérfluo e é identificado quando uma imposição de uma ramificação posterior permite que um ramo ancestral seja removido sem que o valor por ele imposto a alguma variável seja alterado. Convém encolher a árvore com o descarte do nó 2 da fila de nós pendentes. A restrição em x_1 é removida do nó 3 e o problema é reotimizado. Analisando o nó 4 na Figura 2.1B, a restrição $x_3 \geq 1$ ficou folgada, então a árvore é encolhida pela remoção do nó 3 e reotimização no nó 4. Na Figura 2.1C, a solução é inteira com valor 16. A restrição $x_2 \leq 0$ aplicada ao nó 4 leva a uma infactibilidade. Ainda no nó 4, a restrição $x_2 \leq 0$ leva a uma infactibilidade mesmo que a restrição $x_1 \geq 2$ seja removida. Por sua vez, no nó 1, a restrição $x_1 \leq 1$ não leva a uma infactibilidade. Conclui-se que a restrição na variável x_2 é mais influente do que a da variável x_1. Convém então inverter a seqüência dos nós. A idéia aqui é reposicionar os ramos mais influentes o mais próximo possível da raiz da árvore. Na Figura 2.1D, o ramo do nó 1 é devido à restrição $x_2 \geq 1$. E no nó 4, o ramo é associado à restrição $x_1 \geq 2$. No nó 4, falta aplicar a restrição $x_1 \leq 1$ que leva a

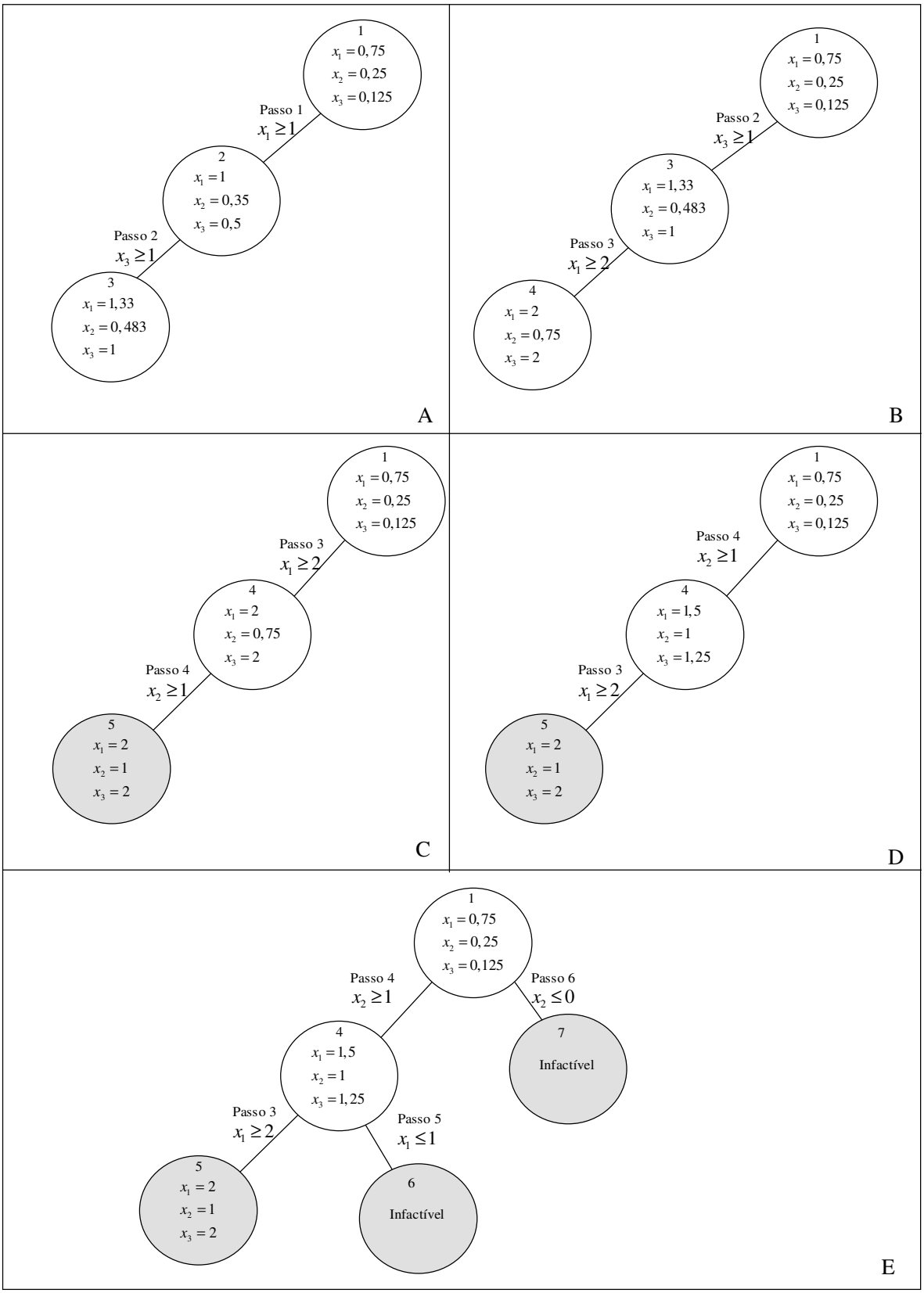

Fig. 2.1: Exemplo de *branch-and-bound* dinâmico.

uma infactibilidade no nó 6 observada na Figura 2.1E. No nó 1, falta adicionar $x_2 \leq 0$ produzindo o nó 7 que é infactível, concluindo assim a enumeração da árvore. Para uma regra de ramificação não revelada em Glover e Tangedahl (1976), o *branch-and-bound* tradicional consumiria 15 nós.

Outra abordagem em que as decisões da árvore podem ser revistas é o *branch-and-bound* paramétrico (Glover, 1978). Aqui um ramo pode parcialmente ou até mesmo completamente desfazer as decisões tomadas em um nó ancestral na árvore de enumeração. Nessa implementação, quando uma solução factível é encontrada para o PIM, ramos não influentes, ou seja, ramos que podem ser removidos sem que a solução do PL corrente seja alterada, são completamente eliminados. O *branch-and-bound* comum impõe apenas restrições de ramificação do tipo $x_j \leq \lfloor \bar{x}_j \rfloor$ ou $x_j \geq \lceil \bar{x}_j \rceil$, com $\bar{x}_j$ representando um valor fracionário assumido na solução ótima do PL num dado nó da árvore para a variável x_j. O *branch-and-bound* paramétrico, por sua vez, estabelece inicialmente as restrições de forma menos rígida pela imposição de penalidades nas restrições de ramificação, alterando para tanto os coeficientes da função objetivo, que passa a ser parametrizada. O modelo matemático (2.1) é relativo à relaxação linear do PIM em que A é um matriz $(m \times n)$, D é uma matriz $(m \times p)$, e é um vetor com todas as componentes iguais a um, x representa o vetor das variáveis 0–1 que foram relaxadas e y é o vetor das variáveis contínuas. Os conjuntos N^- e N^+ representam as variáveis binárias que sofreram ramificações paramétricas, e espera-se que assumam respectivamente os valores 0 e 1 devido à penalidade M_j aplicada para cada $x_j \in N^+ \cup N^-$ na função objetivo. Antes de uma ramificação paramétrica ser consolidada pela substituição por uma ramificação de *branch-and-bound* tradicional, é permitido reavaliar opções de ramificação anteriores e reversões podem ocorrer à maneira apresentada no *branch-and-bound* dinâmico.

$$\begin{aligned} & \min cx + dy + \sum_{j \in N^-} M_j x_j + \sum_{j \in N^+} M_j (1 - x_j) \\ \text{sujeito a} \quad & \\ & Ax + Dy \geq b \\ & e \geq x \geq 0 \\ & y \geq 0 \end{aligned} \tag{2.1}$$

As idéias presentes no *branch-and-bound* dinâmico e no *branch-and-bound* paramétrico envolvem reavaliar a estrutura rígida da busca em árvore, embora o benefício do casamento com uma forma de memória dinâmica e flexível precisou ficar latente até o surgimento da busca tabu.

2.2 Busca tabu

Busca tabu (Glover e Laguna, 1997c) é uma meta-heurística embasada em procedimentos habilitados a transcenderem fronteiras de infactibilidade ou otimalidade local. Restrições são impostas e liberadas para direcionar a ênfase da busca em regiões que seriam evitadas sob a sua ausência. O procedimento é fundamentado no uso de memória adaptativa e na exploração reativa do espaço de busca. A memória adaptativa permite seletivamente armazenar elementos críticos do caminho percorrido e que serão utilizados posteriormente pela exploração reativa, levando a uma análise econômica e efetiva do espaço de busca.

A memória baseada em recentidade, também conhecida como memória de curto prazo, é imprescindível. Com o seu uso, atributos de soluções que foram alterados por movimentos em uma vizinhança definida são considerados tabu e não podem reaparecer nas soluções das próximas iterações. O retorno recorrente e sistemático a soluções visitadas num passado recente é então evitado, fenômeno conhecido como ciclagem.

A memória baseada em freqüência complementa a baseada em recentidade e pode aparecer sob a forma de medidas de transição ou residência registradas pela mudança ou permanência de atributos das soluções visitadas ao longo da busca. Muitas vezes essas medidas são utilizadas para criar penalidades ou incentivos que alteram a medida de avaliação dos movimentos, promovendo assim fases de intensificação ou diversificação, quer pelo incentivo de manter bons atributos, quer pelo incentivo de explorar atributos raramente considerados.

A memória também pode ser usada para avaliar a influência ou o impacto provocado por um movimento, não só pela forma como afeta a qualidade da solução, mas principalmente pela alteração estrutural que pode ser observada na configuração da solução. Armazenar a informação a respeito da influência de escolhas em elementos da solução incorpora um nível a mais de aprendizado.

2.3 Fundamentos da busca tabu paramétrica

Neste trabalho exploramos as idéias da busca tabu paramétrica (BTP), uma heurística genérica para resolver problemas PIM proposta por Glover (2006). A BTP resolve uma série de programas lineares incorporando ramificações de inequações por meio de termos ponderados na função objetivo. A abordagem estende e modifica o *branch-and-bound* paramétrico (Glover, 1978) ao trocar a memória em árvore pelo mecanismo de memória adaptativa da busca tabu, que provê maior flexibilidade e facilita o uso de estratégias que transcendem o escopo da busca em árvore.

2.3.1 Formulação

O problema de programação inteira mista 0–1 é representado por

$$\begin{aligned} &(\text{PIM}) && x_0 = \min cx + dy \\ &\text{sujeito a} \\ & && (x, y) \in W \\ & && x \in X \\ &\text{tal que} \\ & && W = \{(x, y) | Ax + Dy \geq b, e \geq x \geq 0\} \\ & && X = \{e \geq x \geq 0 \text{ e } x \text{ é inteiro}\} \end{aligned} \tag{2.2}$$

No modelo (2.2), A é uma matriz $(m \times n)$, D é uma matriz $(m \times p)$, enquanto e é um vetor com todas as componentes iguais a 1. As desigualdades $Ax + Dy \geq b$ incluem uma restrição associada com a qualidade da função objetivo e tem a forma $cx + dy \leq x_0^* - \varepsilon$, em que x_0^* é o valor da função objetivo da melhor solução conhecida (solução incumbente) do PIM e ε é um número positivo pequeno. A relaxação de programação linear do PIM que contém somente a restrição $(x, y) \in W$é representada por (PL) no modelo (2.3).

$$\begin{aligned} &(\text{PL}) && x_0 = \min cx + dy \\ &\text{sujeito a} \\ & && (x, y) \in W \\ &\text{tal que} \\ & && W = \{(x, y) | Ax + Dy \geq b, e \geq x \geq 0\} \end{aligned} \tag{2.3}$$

O vetor solução x é inteiro-factível se $x \in X$. Uma solução factível para o PIM é uma solução que é PL-factível e inteira-factível.

2.3.2 Princípios básicos da busca tabu paramétrica

Sejam N^+ e N^- subconjuntos de $N = \{1, 2, \ldots, n\}$, o conjunto de índices de x. Considere o conjunto $N' = N^+ \cup N^-$, e seja $x' \in X$ um vetor tentativo, isto é, x' é um vetor parcial que contém componentes x'_j para $j \in N'$, e as componentes $j \in N - N'$ não são consideradas. A BTP procura impor as seguintes condições:

$$x_j \geq x'_j, \quad j \in N^+ \quad \text{se} \quad (x'_j = 1) \qquad (UP) \tag{2.4}$$

$$x_j \leq x'_j, \quad j \in N^- \quad \text{se} \quad (x'_j = 0) \qquad (DN) \tag{2.5}$$

As condições "cima" (*UP*) e "baixo" (*DN*) das expressões (2.4) e (2.5) representam *metas* e x'_j o valor da *meta*. Tais condições não são estabelecidas diretamente como no *branch-and-bound* tradicional, mas sim indiretamente ao serem incluídas por meio de pesos na função objetivo do PL. O problema parametrizado[1] resultante de programação linear, em que M_j é um número positivo suficientemente grande, é dado por (2.6).

$$\begin{aligned} &\text{(PL')} \qquad u_0 = \min cx + dy + \sum_{j \in N^-} M_j x_j + \sum_{j \in N^+} M_j (1 - x_j) \\ &\text{sujeito a} \\ &\qquad\qquad (x, y) \in W \\ &\text{tal que} \\ &\qquad\qquad W = \{(x, y) | Ax + Dy \geq b, e \geq x \geq 0\} \end{aligned} \tag{2.6}$$

Uma abordagem de duas fases é usada para a resolução do PL', ao criar uma fase inicial com a função objetivo $\left(\sum_{j \in N^-} M_j x_j + \sum_{j \in N^+} M_j (1 - x_j) \right)$ e descartando a componente $cx + dy$. Na solução ótima da fase I, todas as variáveis não-básicas são mantidas em seus limitantes (inferior ou superior) e removidas do problema. Na segunda fase, minimiza-se $cx + dy$ sobre o problema residual.

Embora em uma abordagem simples, todos os M_j possam ser iguais a 1 na fase I, é mais interessante escolhê-los de forma a colocar maior ênfase nas metas mais recentemente estabelecidas. Isto pode ser conseguido com uso de uma função com decaimento exponencial proposta por Glover (2006), em que a cada iteração, as variáveis com metas têm seus pesos atualizados e perdem influência à medida que as iterações avançam.

$$M_j = \begin{cases} \left[(1 + r)^{NI - (Iter - GI_j)}\right] & \text{se} \quad \boldsymbol{NI - (Iter - GI_j) \geq 0} \\ 1 & \text{c.c.} \end{cases} \tag{2.7}$$

Na expressão (2.7), $r > 0$, $\boldsymbol{GI_j}$ é a iteração em que uma meta foi estabelecida para a variável binária x_j, $Iter$ é a iteração corrente e *NI* controla o número de iterações em que a memória do peso persiste. O operador $[\cdot]$ arredonda um número fracionário para o inteiro mais próximo. Além disso, quanto maior o valor de r, menor será a influência de pesos estabelecidos em iterações mais antigas.

Duas instâncias de PL' diferem somente pela função objetivo. Portanto, a partir da solução ótima

[1] Em geral, essa técnica de somar as restrições na função objetivo é conhecida como penalização, mas mantivemos a terminologia *paramétrica* proposta por Glover (1978).

da primeira instância obtém-se a solução ótima da segunda instância através de reotimização pelo método primal simplex aplicado às duas fases.

O método de busca tabu paramétrica é iniciado com uma instância de PL' correspondente à relaxação original, quando N' é vazio. Uma solução ótima de uma instância de PL' é representada por (x'', y'') e como estamos interessados no valor ótimo das variáveis binárias, fazemos referência a x'' como uma solução ótima do PL', e y'' torna-se implícito. O método de busca tabu paramétrica prossegue com o uso da informação da solução do PL', incluindo a relação entre x'' e x' para determinar um novo vetor x' e um problema associado PL'. Quando a solução de um PL' é inteira-factível, a restrição de função objetivo é atualizada com o novo x_0^* e o método dual simplex é usado para a reotimização. Este processo é repetido até atingir um critério de parada.

2.3.3 Transições PL'

Transições de uma instância de PL' para outra instância são baseadas em regras que redefinem os valores das metas em x'. Estas transições são expressas como:

(i) Faça $x'_j := \lfloor x''_j \rfloor + 1$ e inclua j em N^+ (nova meta $x_j \geq x'_j$) (*T-UP*)

(ii) Faça $x'_j := \lceil x''_j \rceil - 1$ e inclua j em N^- (nova meta $x_j \leq x'_j$) (*T-DN*)

(iii) Remova j de N' (libera x_j de (*UP*) e (*DN*)) (T-*LIVRE*)

Note que as transições (*T-UP*) e (*T-DN*) são geradas a partir do piso e do teto da solução x''_j do PL'. A transição (*T-LIVRE*) faz com que uma variável x_j fique livre de (*UP*) e (*DN*). A execução destas transições para um conjunto apropriado de variáveis depende de dois tipos de condições descritas a seguir.

2.3.4 Infactibilidade direta de metas

Uma solução ótima $x = x''$ de um PL' é dita diretamente infactível com relação a uma meta se viola uma das condições *UP* ou *DN*, isto é,

(i) para algum $j \in N^+, x''_j < x'_j$ (*V-UP*)

(ii) para algum $j \in N^-, x''_j > x'_j$ (*V-DN*)

Em uma violação (*V-UP*) ou (*V-DN*), diz-se que a variável x_j é diretamente infactível com relação à meta e define-se então o conjunto $G = \{j \in N' \mid x_j$ é infactível com relação à meta$\}$. A *resposta primária* à infactibilidade direta de metas consiste em estabelecer novas metas no sentido oposto para um subconjunto G_p selecionado de G, tal que $G_p = \{j \in G \mid \text{resposta}(R{-}UP)$ ou $(R{-}DN)$ é realizada$\}$.

Se $x''_j < x'_j$ para $j \in N^+$, transfira j de N^+ para N^- e faça $x'_j := \lceil x''_j \rceil - 1$ (*R-DN*)

Se $x''_j > x'_j$ para $j \in N^-$, transfira j de N^- para N^+ e faça $x'_j := \lfloor x''_j \rfloor + 1$ (*R-UP*)

Além de respostas primárias, consideramos (*R-LIVRE*) uma resposta secundária à infactibilidade de meta que consiste em tornar livre uma variável que pertence a G. Associamos então um subconjunto selecionado G_s de G, tal que G_s = {$j \in G$ | resposta (*R-LIVRE*) é realizada}.

2.3.5 Resistência direta à meta

A resistência direta à meta $GR_j(UP)$ ou $GR_j(DN)$ da variável $x_j, j \in G$, representa o nível que a violação (*V-UP*) ou (*V-DN*) resiste à imposição da meta correspondente (*UP*) ou (*DN*), ou de forma mais geral, a variação na função objetivo do PL' causada pela resposta (*R-DN*) ou (*R-UP*) respectivamente.

Partindo-se do princípio de que valores altos de resistência causam um impacto maior que valores baixos de resistência, temos o conjunto G_p composto das g_p variáveis com maiores valores de resistência à meta do conjunto G, enquanto que o conjunto G_s é composto das g_s com maiores valores de resistência à meta do conjunto $G - G_p$. Na presença de infactibilidade de metas escolhemos sempre $g_p \geq 1$, mas pode-se adotar $g_s = 0$. A criação dos conjuntos G_p e G_s e as possibilidades de decisões táticas para o cálculo de GR_j que foram utilizadas são apresentadas nas subseções 2.3.12 e 2.3.13.

2.3.6 Infactibilidade potencial de meta

Seja x_j uma variável cuja condição de meta é correntemente satisfeita, $x''_j = x'_j$. Se x_j tem seu valor M_j reduzido em uma quantidade pequena, então esta condição de meta pode tornar-se violada. Variáveis que apresentam esse comportamento são chamadas de *potencialmente infactíveis* com relação às metas.

Para tratar a infactibilidade potencial de metas, define-se uma medida de resistência à meta, representada por GR^0_j, relacionada com o decréscimo em M_j para tornar uma variável x_j infactível com relação à sua meta. O critério adotado foi estabelecer $GR^0_j = -CR_j$, em que CR_j é o custo reduzido da variável x_j associado com a solução ótima do PL'. Na otimalidade do PL', $CR_j \geq 0$[2], $j \in N'$ e $CR_j = 0$ para toda variável x_j com resistência direta à meta, por ser básica. As variáveis potencialmente infactíveis com relação a metas são então aquelas com metas atendidas e que apresentam os maiores valores de GR^0_j. As mesmas respostas primárias e secundárias aplicam-se a estas variáveis, ao substituir (*V-UP*) e (*V-DN*) por (*V-UP*^{0}) e (*V-DN*^{0}) conforme as expressões (2.8) e (2.9), nas quais

[2]Uma variável não-básica no limitante superior 1 tem custo reduzido não-positivo, mas por convenção, pode-se referir ao custo reduzido não-negativo da variável de folga $s_j = 1 - x_j$.

T' representa um limiar de aceitação para classificar uma variável como potencialmente infactível em relação a sua meta.

$$\text{Para algum } j \in N^+, \quad x'' = x' \text{ e } GR_j^0 \geq T' \qquad (V{-}UP^0) \tag{2.8}$$

$$\text{Para algum } j \in N^-, \quad x'' = x' \text{ e } GR_j^0 \geq T' \qquad (V{-}DN^0) \tag{2.9}$$

Ao contrário de estabelecer explicitamente o valor T', adotou-se um parâmetro T^0 que limita o número máximo de variáveis aceitas como potencialmente infactíveis em relação a suas metas. Todas as variáveis com metas atendidas são ordenadas em ordem decrescente de valor GR_j^0 e as T^0 variáveis com maiores valores são aceitas como potencialmente infactíveis em relação a suas metas. Esse parâmetro T^0 corresponde a um valor implícito da menor medida de GR_j^0 para uma variável ser aceita como potencialmente infactível. Essas variáveis farão parte do conjunto G, e dependendo de como os conjuntos G_p e G_s são definidos, terão suas metas invertidas ou liberadas, respectivamente.

Variáveis diretamente infactíveis com relação a metas são consideradas mais importantes que variáveis potencialmente infactíveis com relação a metas. Portanto, no ordenamento das variáveis pelos seus valores de resistência, os valores de resistência a metas das variáveis diretamente infactíveis são maiores que os valores das variáveis potencialmente infactíveis com relação a metas. O principal uso deste último tipo de variável ocorre na situação em que todas as variáveis diretamente infactíveis com relação a metas são excluídas por um critério de busca tabu.

2.3.7 Infactibilidade inteira

A infactibilidade inteira é considerada apenas quando não ocorre a infactibilidade de metas e a solução $x = x''$ atribui um valor fracionário a algum componente x_j de x. Seja o conjunto $F = \{j \in N \mid x_j = x''_j \text{ é fracionário}\}$ de todas as variáveis que assumem valores fracionários na solução ótima de um PL'. Dizemos que uma variável x_j é fracionária irrestrita se x_j é fracionária, mas não infactível com relação à meta, e portanto é incluída no conjunto D, tal que $D = F - G$ e $d = |D|$. Não existe resposta primária específica para estas variáveis, e a decisão entre (*R-UP*) e (*R-DN*) é estabelecida pela preferência de escolha para cada variável, descrita a seguir.

2.3.8 Penalidade inteira e medidas de preferência de escolha

A fim de efetuar uma transição (*T-UP*) ou (*T-DN*) para uma variável fracionária irrestrita, são criadas medidas de penalidade inteira $IP_j(UP)$ e $IP_j(DN)$ correspondentes, que refletem o grau de deterioração da função objetivo do PL' que é induzido pela imposição da restrição associada $x_j \geq \lceil x''_j \rceil$ ou $x_j \leq \lfloor x''_j \rfloor$. Além disso, uma medida de escolha preferencial baseada tanto em $IP_j(UP)$ quanto

$IP_j(DN)$ é necessária para classificar as variáveis irrestritamente fracionárias quanto à preferência de escolha para receberem transições. As d_0 variáveis com maiores valores de escolha preferencial são incluídas no conjunto $D_0 \subset D$ e são as escolhidas para receberem metas. Três medidas de escolha preferencial foram adotadas:

$$CP_j = [IP_j(UP) + IP_j(DN)]\,[|IP_j(UP) - IP_j(DN)| + w] \tag{2.10}$$

$$CP_j = (1-\mu)\min\{IP_j(DN), IP_j(UP)\} + \mu\max\{IP_j(DN), IP_j(UP)\} \tag{2.11}$$

$$\begin{array}{ll} CP_j = IP_j(UP) & \text{se} \quad f_j^+ \leq f_j^- \\ CP_j = IP_j(DN) & \text{se} \quad f_j^+ > f_j^- \end{array} \tag{2.12}$$

Na expressão (2.10) proposta por Glover (2006), w é um valor positivo pequeno para contornar o caso em que a diferença das penalidades é igual a zero e na expressão (2.11), (Eckestein, 1994), $0 \leq \mu \leq 1$. Em (2.12), aqui proposta, $f_j^+ = \lceil x_j'' \rceil - x_j''$ e $f_j^- = x_j'' - \lfloor x_j'' \rfloor$, sendo x_j'' o valor de x_j na solução ótima do PL'. Uma vez definido o conjunto D_0, as respostas são assim estabelecidas por (2.13) e (2.14):

$$\text{Se } IP_j(DN) \leq IP_j(UP), \quad j \in D, \quad \text{execute } (T{-}DN) \qquad (R{-}DN^D) \tag{2.13}$$

$$\text{Se } IP_j(UP) < IP_j(DN), \quad j \in D, \quad \text{execute } (T{-}UP) \qquad (R{-}UP^D) \tag{2.14}$$

As táticas utilizadas para calcular $IP_j(UP)$, $IP_j(DN)$ e definir a cardinalidade do conjunto D_0 são apresentadas nas subseções 2.3.12 e 2.3.13.

2.3.9 Condição tabu e atualização da duração tabu

Uma restrição tabu é associada a uma resposta (*R-DN*) ou (*R-UP*) de uma variável x_j proibindo sua execução se a resposta oposta ((*R-UP*) ou (*R-DN*), respectivamente) foi executada para x_j dentro das últimas iterações, de acordo com a duração tabu. Para simplificar a apresentação, dizemos que (*R-DN*) e (*R-UP*) também fazem referência às respostas de variáveis potencialmente infactíveis (*R-DN*^{0}) e (*R-UP*^{0}). O valor da duração tabu associado a uma variável x_j depende da história da busca e não é imposta sob infactibilidade inteira, apenas sob infactibilidade de metas. *DuraçãoTabu$_j$(UP)* e *DuraçãoTabu$_j$(DN)* denotam a duração tabu de uma variável x_j se a restrição foi ativada por uma (*R-DN*) ou (*R-UP*), respectivamente. Similarmente, *TabuFim$_j$*(UP) e *TabuFim$_j$*(DN) representam a iteração em que a duração tabu da variável x_j termina. Se uma resposta (*R-UP*) ocorrer na iteração *Iter*

então a restrição tabu (2.15) é estabelecida para proibir que a resposta oposta (*R-DN*) seja realizada pelo período de $Random\,(\textit{DuraçãoTabu}_j\textit{(DN)}, \textit{MaxDuração})$[3] iterações. Analogamente, se uma resposta (*R-DN*) ocorrer na iteração *Iter*, então a restrição tabu (2.16) é estabelecida para proibir que a resposta oposta (*R-UP*) seja realizada pelo período de $Random\,(\textit{DuraçãoTabu}_j\textit{(UP)}, \textit{MaxDuração})$ iterações.

$$TabuFim_j(DN) = Iter + Random\,(\textit{DuraçãoTabu}_j\textit{(DN)}, \textit{MaxDuração}) \tag{2.15}$$

$$TabuFim_j(UP) = Iter + Random\,(\textit{DuraçãoTabu}_j\textit{(UP)}, \textit{MaxDuração}) \tag{2.16}$$

2.3.10 Critério de aspiração por resistência

A aspiração por resistência é baseada na maior resistência de uma resposta gerada no passado. Assim, $\textit{Aspiração}_j(DN)$ e $\textit{Aspiração}_j(UP)$ são os maiores valores de resistência a metas $GR_j(DN)$ e $GR_j(UP)$ que ocorreram para uma variável x_j em alguma iteração em que esta variável foi selecionada para executar uma resposta (*R-UP*) ou (*R-DN*), respectivamente. A restrição tabu é desconsiderada para uma resposta (*R-UP*) se

$$GR_j(DN) > \textit{Aspiração}_j(UP),$$

e a restrição tabu é desconsiderada para uma resposta (*R-DN*) se

$$GR_j(UP) > \textit{Aspiração}_j(DN).$$

Uma resposta é dita admissível se não é tabu ou se satisfaz o critério de aspiração, e chamada inadmissível, caso contrário. Se a resposta para uma variável infactível com relação à meta é inadmissível, então a variável não pode entrar nos conjuntos G_p ou G_s. A única exceção a esta regra ocorre quando G_p é vazio em virtude de restrições tabu. Neste caso utiliza-se o critério de aspiração por exclusão (*default*) em que a variável com menor número de iterações para se tornar não-tabu é escolhida para entrar em G_p.

Da mesma forma em que variáveis diretamente infactíveis dominam as variáveis potencialmente infactíveis, as variáveis diretamente infactíveis com relação a metas e associadas à $\textit{Aspiração}_j$ dominam as variáveis potencialmente infactíveis associadas à $\textit{Aspiração}_j^0$, que é definida a partir de GR_j^0.

[3] O operador $Random(a, b)$ sorteia um número inteiro aleatório no intervalo $[a, b]$ com distribuição uniforme.

2.3.11 Determinação dos valores de *DuraçãoTabu*$_j$

Adota-se aqui a notação em que α se refere tanto à condição (*UP*) quanto (*DN*) e β se refere à condição oposta (*DN*) e (*UP*) respectivamente. Sempre que a reação $(R-\alpha)$ é criada, $(R-\beta)$ recebe uma restrição tabu que permanecerá por $Random\,(\textit{DuraçãoTabu}_j(\beta), \textit{MaxDuração})$ iterações. As medidas *DuraçãoTabu*$_j(\beta)$ são dinamicamente atualizadas com o seguinte procedimento proposto por Glover (2006) e complementado aqui com as expressões para duração tabu e min/max:

1. Os valores de *DuraçãoTabu*$_j(\alpha)$ iniciam em um valor mínimo *MinDuração* $= k_1 \times m$, sendo m o número de restrições do PL', $0 < k_1 \leq 1$ e *Aspiração*$_j(\alpha) := -\infty$ para $j \in N$.

2. Sempre que x_j receber uma reação $(R-\alpha)$, devido a uma violação $(V-\beta)$, incrementa-se *DuraçãoTabu*$_j(\beta)$ em um valor especificado tal que *DuraçãoTabu*$_j(\beta) :=$ *DuraçãoTabu*$_j(\beta)+$ $\max(1, k_2 \times n)$, sendo n o número de variáveis binárias e $0 < k_2 \leq 1$. A *DuraçãoTabu*$_j(\beta)$ não é alterada se a reação é tabu e é executada por satisfazer um critério de aspiração. O valor *DuraçãoTabu*$_j(\beta)$ não pode ultrapassar um valor máximo *MaxDuração* $= k_3 \times m$, com $0 < k_1 \leq k_3 \leq 1$.

3. Após *MaxLimpaTabuIter* iterações, e toda vez que uma nova solução inteira-factível é encontrada, o processo é completamente reiniciado. Todas as durações tabu são novamente definidas como *MinDuração* e todos os valores *Aspiração*$_j(\alpha) := -\infty$.

2.3.12 Medidas de resistência a metas e penalidade inteira

Três mecanismos de cálculo de resistência a metas e de penalidade inteira foram implementados. No nível mais simples, Glover (2006) sugere que GR_j pode ser a distância à meta (DM), ou a diferença absoluta $|x''_j - x'_j|$, que identifica quão longe o valor x''_j da solução do PL' está da sua meta atual x'_j. Para o cálculo da penalidade inteira, após encontrar as frações $f_j^+ = \lceil x''_j \rceil - x''_j$ e $f_j^- = x''_j - \lfloor x''_j \rfloor$, usa-se $IP_j(UP) = f_j^+$, $IP_j(DN) = f_j^-$ e CP_j é computado com a expressão (2.10).

No início da busca, toda nova meta estabelecida provoca uma deterioração no valor da função objetivo, pois sempre que o PL' é infactível inteiro, na fase II do PL' alguma variável que estava livre é fixada no limitante superior ou inferior, tornando o problema mais restrito. Porém, quando a infactibilidade de metas é atingida, a inversão de um conjunto de metas pode levar a uma melhoria no valor da função objetivo em relação ao PL' corrente. Assim, em um nível mais sofisticado, Glover (2006) sugere que as pseudopenalidades da programação inteira podem ser usadas, e então GR_j pode ser uma estimativa da variação na função objetivo do PL' causada pela execução da resposta (*R-DN*) ou (*R-UP*). A técnica de ramificação *Reliability Branching* (Achterberg et al., 2005) que é uma generalização do pseudocusto com inicialização por *strong branching* usa esse princípio. Como alcançou

bons resultados em algoritmos de *branch-and-cut* para instâncias da Miplib 2003 e instâncias mantidas por Mittelmann (2003), foi adaptada para os cálculos de resistência a meta e de penalidade inteira no contexto da BTP.

No início da busca, enquanto os pseudocustos disponíveis ψ_j^- e ψ_j^+ ainda não são confiáveis, o *Reliability Branching* toma decisões ao estilo do *strong branching*, mas após uma variável ter sido escolhida várias vezes para sofrer ramificação, o seu pseudocusto se torna confiável e passa a ser usado, evitando assim reotimizações de PL's. Ao final da fase I, as variáveis com metas estabelecidas e atendidas foram fixadas. Já as variáveis binárias x_j cujos valores correntes são fracionários e que ainda não possuem pseudocustos confiáveis precisam ser analisadas. Sejam então Q um PL' que apresenta infactibilidade inteira, Q_j^- e Q_j^+ os dois subproblemas derivados quando em Q são impostas as restrições explícitas $x_j \leq \lfloor x_j'' \rfloor$e $x_j \geq \lceil x_j'' \rceil$ respectivamente. Seja x_0 o valor ótimo da função objetivo para Q. Por sua vez, $x_{Q_j^-}$ e $x_{Q_j^+}$ são os valores de função objetivo subótimos ao se aplicar um máximo de γ iterações duais simplex a Q_j^- e Q_j^+. As variações no valor da função objetivo são respectivamente $\tilde{\Delta}_j^- = x_{Q_j^-} - x_0$ e $\tilde{\Delta}_j^+ = x_{Q_j^+} - x_0$. A medida CP_j é calculada com a expressão (2.11). Os pseudocustos de uma variável x_j são definidos por $\psi_j^+ = \{\sum_{k=1}^{\eta_j^+} (\tilde{\Delta}_{j,k}^+ / f_{j,k}^+)\}/\eta_j^+$ e $\psi_j^- = \{\sum_{k=1}^{\eta_j^-} (\tilde{\Delta}_{j,k}^- / f_{j,k}^-)\}/\eta_j^-$. Os valores η_j^+ e η_j^- indicam quantas vezes os pseudocustos ψ_j^+ e ψ_j^- foram atualizados com Q_j^+ e Q_j^- factíveis, $f_j^+ = \lceil x_j'' \rceil - x_j''$ e $f_j^- = x_j'' - \lfloor x_j'' \rfloor$. As penalidades *UP* e *DN* são respectivamente $IP_j(UP) = f_j^+ \psi_j^+$ e $IP_j(DN) = f_j^- \psi_j^-$. Se um dos dois subproblemas for infactível, utiliza-se o pseudocusto ψ_j^- ou ψ_j^+ atual da variável, mesmo que ainda não seja confiável. Porém, se essa variável for escolhida para receber meta, o lado infactível nunca é escolhido por corresponder a uma meta que não será atendida na próxima iteração. Caso ambos os problemas sejam infactíveis, então $CP_j = -\infty$, colocando assim prioridade mínima na escolha dessa variável. A Figura 2.2 mostra o algoritmo *Reliability Branching* para infactibilidade inteira. Os pseudocustos de uma variável x_j são ditos não-confiáveis se $\min\{\eta_j^-, \eta_j^+\} < \eta_{rel}$. Os parâmetros μ da expressão (2.11), η_{rel} que determina quando um pseudocusto é confiável, λ presente no (*Passo* 2 *d*) e γ são determinados empiricamente.

O cálculo de resistência a metas apresenta uma implementação similar conforme a Figura 2.3. Ao contrário de calcular $IP_j(UP)$, $IP_j(DN)$ e CP_j, calcula-se na inicialização do *Passo* 2, $GR_j(UP) = f_j^+ \psi_j^+$ ou $GR_j(DN) = f_j^- \psi_j^-$, e no *Passo* 2*c*, $GR_j(UP) = \tilde{\Delta}_j^+$ ou $GR_j(DN) = \tilde{\Delta}_j^-$, conforme o caso. Além disso, no lugar de D e D_0, os conjuntos considerados são G, G_p e G_s. No *Passo* 2*d* e no *Passo* 3 , $GR_j(UP/DN)$ indica que apenas uma das medidas $GR_j(UP)$ ou $GR_j(DN)$ é cálculada.

A terceira opção de cálculo de resistência a metas e penalidade inteira é uma variação do esquema $\mathbf{P}^4$ da *Active-Constraint Variable Ordering* (ACVO) de Patel e Chinneck (2007). Esse mecanismo

[4]Todos os outros esquemas de **A** a **O** presentes em Patel e Chinneck (2007), foram implementados e testados no

Passo 1	Considere o conjunto D das variáveis fracionárias irrestritas.
Passo 2	Ordene D usando CP_j, em ordem não crescente, com $IP_j(UP) = f_j^+\psi_j^+$ e $IP_j(DN) = f_j^-\psi_j^-$. Para cada $j \in D$ com $\min\{\eta_j^-, \eta_j^+\} < \eta_{rel}$ faça:
	a) Realize γ iterações duais simplex em Q_j^- e Q_j^+ e encontre os valores $\tilde{\Delta}_j^-$ e $\tilde{\Delta}_j^+$, representando os acréscimos no valor da função objetivo.
	b) Atualize os pseudocustos ψ_j^- e ψ_j^+ com $\tilde{\Delta}_j^-$ e $\tilde{\Delta}_j^+$.
	c) Atualize os valores de preferência de escolha CP_j, com $IP_j(UP) = \tilde{\Delta}_j^+$ e $IP_j(DN) = \tilde{\Delta}_j^-$.
	d) Se $\max_{j\in D}\{CP_j\}$ não é alterado por λ atualizações consecutivas, vá ao *Passo* 3.
Passo 3	Retorne os d_0 índices $j \in D$ com maiores valores de CP_j para compor o conjunto D_0.

Fig. 2.2: *Reliability Branching* para infactibilidade inteira.

de seleção avalia a influência de uma variável em uma restrição ativa e vice-versa. Ao contrário do pseudocusto que avalia o grau de deterioração na função objetivo, a ACVO prioriza a descoberta de soluções inteiras o mais rápido possível, sendo bastante conveniente para a BTP. Toda variável inteira x_j que assume um valor fracionário recebe um peso w_{ij} associado a cada uma das m restrições de acordo com a expressão (2.17). Nessa expressão, N_i^F representa o número de variáveis binárias que assumem valor fracionário na restrição ativa i e a_{ij} são os coeficientes da matriz A. A motivação de (2.17) é baseada na idéia de que um valor grande de a_{ij} apresenta um impacto maior na restrição ativa, enquanto que uma grande quantidade de variáveis N_i^F tende a minimizar este impacto. Se x_j não aparece na restrição i ou se a restrição i não for ativa, $w_{ij} = 0$. As variáveis com maior soma total de pesos w_{ij} devem ser as escolhidas, g_p e g_s variáveis, se o PL' for infactível nas metas ou d_0 variáveis, se o PL' for infactível inteiro. No contexto da busca tabu paramétrica, para que a ACVO se torne mais eficiente, verificou-se empiricamente que é mais eficaz multiplicar os pesos w_{ij} pelo termo $1/f_j^+$ ou $1/f_j^-$ conforme a expressão (2.18) proposta nesta tese. Note que multiplicar por $1/f_j^+$ e $1/f_j^-$ em (2.18) favorece a escolha das variáveis mais próximas da integralidade.

$$w_{ij} = \begin{cases} \frac{|a_{ij}|}{N_i^F} & \text{se a variável } x_j \text{ aparece na restrição ativa } i \\ 0 & \text{caso contrário} \end{cases} \tag{2.17}$$

$$\begin{aligned} IP_j(DN) = \infty, \quad IP_j(UP) = \tfrac{1}{f_j^+}\sum_{i=1}^{m} w_{ij} \quad \text{se} \quad f_j^+ \leq f_j^- \\ IP_j(UP) = \infty, \quad IP_j(DN) = \tfrac{1}{f_j^-}\sum_{i=1}^{m} w_{ij} \quad \text{se} \quad f_j^+ > f_j^- \end{aligned} \tag{2.18}$$

contexto da BTP, porém com resultados pouco promissores.

Passo 1 Considere o conjunto G.

Passo 2 Ordene G usando $GR_j(UP) = f_j^+\psi_j^+$ ou $GR_j(DN) = f_j^-\psi_j^-$, em ordem não crescente. $GR_j(UP)$ é usado se a variável de índice j apresenta V-UP e $GR_j(DN)$ é usado se apresenta V-DN.
Para cada $j \in G$ com $\min\{\eta_j^-, \eta_j^+\} < \eta_{rel}$ faça:

a) Se j está associado a V-UP, realize γ iterações duais simplex em Q_j^+ e encontre o valor $\tilde{\Delta}_j^+$ representando a variação no valor da função objetivo. Caso contrário, realize γ iterações duais simplex em Q_j^- e encontre o valor $\tilde{\Delta}_j^-$.

b) Atualize o pseudocusto ψ_j^+ com $\tilde{\Delta}_j^+$ se no *Passo* 2*a* foi calculado $\tilde{\Delta}_j^+$ e atualize ψ_j^- com $\tilde{\Delta}_j^-$ caso contrário.

c) Atualize o valor de resistência a meta, $GR_j(UP) = \tilde{\Delta}_j^+$ ou $GR_j(DN) = \tilde{\Delta}_j^-$ se no *Passo* 2*a* foi calculado $\tilde{\Delta}_j^+$ ou $\tilde{\Delta}_j^-$ respectivamente.

d) Se $\max_{j\in G}\{GR_j(UP/DN)\}$ não é alterado por λ atualizações consecutivas, vá ao *Passo* 3.

Passo 3 Retorne os g_p índices $j \in G$ com maiores valores de $GR_j(UP/DN)$ para compor o conjunto G_p e os g_s índices $j \in G - G_p$ com maiores valores de $GR_j(UP/DN)$ para compor o conjunto G_s.

Fig. 2.3: *Reliability Branching* para infactibilidade de metas.

A medida CP_j é calculada com (2.12). Assim, se uma variável está mais próxima do teto, $IP_j(UP)$ é considerada e *T-UP* será realizada pela expressão (2.13) pois $IP_j(DN) = \infty$. E se estiver mais próxima do piso *T-DN*, será realizada por (2.14), pois $IP_j(UP) = \infty$. Os valores de resistência a metas são calculados de maneira similar com as expressões (2.19) e (2.20).

$$GR_j(UP) = \frac{1}{f_j^+}\sum_{i=1}^{m} w_{ij} \tag{2.19}$$

$$GR_j(DN) = \frac{1}{f_j^-}\sum_{i=1}^{m} w_{ij} \tag{2.20}$$

2.3.13 Cardinalidade dos conjuntos G_p, G_s e D_0

Uma forma simples de determinar a cardinalidade dos conjuntos G_p, G_s e D_0 envolve o uso de frações f_p, f_s e f_d, tais que:

$$|D_0| = d_0 = \max\left(d_{\min}, f_d \times |D|\right), \quad 0 < f_d \leq 1 \tag{2.21}$$

$$|G_p| = g_p = \max\left(g_{p_{\min}}, f_p \times |G|\right), \quad 0 < f_p \leq 1 \tag{2.22}$$

$$|G_s| = g_s = \max\left(g_{s_{\min}}, f_s \times |G|\right), \quad 0 < f_s \leq 1 \tag{2.23}$$

Há um limiar mínimo de cardinalidades aceitáveis, $d_{\min}$, $g_{p_{\min}}$ e $g_{s_{\min}}$ que respeitam $d_{\min} \leq |D|$ e $g_{p\min} + g_{s\min} \leq |G|$. A fragilidade dessa abordagem reside em considerar frações f_p, f_s e f_d, e valores $d_{\min}$, $g_{p_{\min}}$ e $g_{s_{\min}}$ que não são alterados com a evolução da busca. Neste trabalho é proposto um segundo critério que ajusta as cardinalidades de forma adaptativa conforme as características da região sendo visitada e é apresentado na Figura 2.4. O *Passo* 0 do algoritmo envolve a inicialização das variáveis envolvidas, $Iter$ é a iteração corrente, $IterTrans$ é a iteração em que alguma transição ocorreu, e pode ser a transição da infactibilidade inteira para a infactibilidade de metas e vice-versa, ou a permanência por um certo número de iterações na infactibilidade de metas ou infactibilidade inteira. A variável $flagI$ é verdadeira se a busca se encontra na infactibilidade inteira, e falsa, caso contrário. A variável booleana $prox\_g_s$ controla o incremento em g_s e g_p.

O *Passo* 1 trata a infactibilidade inteira. Dados $\delta_2 > 1$, $k_4 \leq 1$, $0 < f_{d_{\max}} \leq 1$ e $f_s < 1$, f_d é incrementado com $f_d := \min(f_{d_{\max}}, f_d \times \delta_2)$ sempre que a busca se mantém por $k_4 \times m$ iterações consecutivas na infactibilidade inteira. Note que f_d não pode superar o valor máximo permitido $f_{d_{\max}}$. Além disso, uma vez que $0 < \delta_1 < 1$, f_d é diminuído, no *Passo* 3 com $f_d := (f_d \times \delta_1)$. Isso ocorre sempre que a busca transita da infactibilidade inteira para a infactibilidade de metas. O raciocínio por trás dessa política é começar a busca estabelecendo simultaneamente várias metas por iteração enquanto a busca está na infactibilidade inteira. Toda vez que a busca transita para a infactibilidade de metas, f_d é abruptamente reduzido, por exemplo $\delta_1 = 0,5$. Quando a busca retornar à infactibilidade de metas, encontrará um valor de f_d bem menor. Por outro lado, a permanência de um número razoável de iterações consecutivas na infactibilidade inteira, $k_4 \times m$, serve de estímulo para aumentar um pouco valor de f_d, com por exemplo $\delta_2 = 1,05$.

O *Passo* 2 é o responsável pelo tratamento da infactibilidade de metas, tipicamente $g_{p_{\min}} = 1$ e $g_{s_{\min}} = 0$. Portanto, quando a busca entra na infactibilidade de metas, procura-se resolver o conflito alterando o mínimo possível as metas estabelecidas. Nessa situação, a cada iteração, apenas uma meta é invertida e nenhuma é removida, pois $g_p = 1$ e $g_s = 0$. Com a permanência continuada na infactibilidade de metas, a cada $k_4 \times m$ iterações, g_s e g_p são nessa ordem, alternadamente incrementados em uma unidade com o auxílio da variável booleana $prox\_g_s$. Caso g_p ou g_s atinja o valor $g_{\max}$ e a infactibilidade de metas não seja resolvida nas $k_4 \times m$ iterações seguintes, o objetivo passa a ser liberar um número grande de variáveis, adota-se $g_p = 1$ e g_s passa a ser calculado de acordo com (2.23) considerando $g_{s_{\min}} = g_{\max}$, ou seja, $g_s = \max(g_{\max}, f_s \times |G|)$. Toda vez que o problema

Passo 0 Inicialização.

$g_p := g_{p_{\min}}$, $g_s := g_{s_{\min}}$, $f_d := f_{d_{\max}}$, $IterTrans := 0$, $Iter := 0$, $flagI := true$ **e** $prox\_g_s := false$

Passo 1 Busca está na infactibilidade inteira.

1.1 se $(Iter - IterTrans) > k_4 \times m$ **então** $f_d := \min(f_{d_{\max}}, f_d \times \delta_2)$ **e**
$IterTrans := Iter$
1.2 Vá para o *Passo* 3

Passo 2 Busca está na infactibilidade de metas.

2.1 se $(Iter - IterTrans) > k_4 \times m$ **então**
$IterTrans := Iter$ **e**
se $(g_p \geq g_{\max})$ **ou** $(g_s \geq g_{\max})$ **então** $g_p := 1$ **e** $g_s = \max(g_{\max}, f_s \times |G|)$
senão
se $prox\_g_s = true$ **então** $g_s := g_s + 1$ **e** $prox\_g_s := false$
senão $g_p := g_p + 1$ **e** $prox\_g_s := true$
2.2 Vá para o *Passo* 3

Passo 3 Construção do novo PL'.

Crie as novas metas, atualize a lista tabu caso o PL' da iteração atual seja infactível nas metas e resolva o novo PL' e faça $Iter := Iter + 1$.
se o novo PL' for infactível nas metas **então**
se $flagI = true$ **então** $flagI := false$, $f_d := (f_d \times \delta_1)$, $prox\_g_s := true$ **e**
$IterTrans := Iter$
Vá ao *Passo* 2
senão
se $flagI := false$ **então** $flagI := true$, $g_p := g_{p_{\min}}$, $g_s := g_{s_{\min}}$ **e**
$IterTrans := Iter$
Vá ao *Passo* 1

Fig. 2.4: Controle adaptativo dos conjuntos G_p, G_s e D_0.

sai da infactibilidade de metas, g_p e g_s retornam respectivamente aos valores iniciais $g_{p_{\min}}$ e $g_{s_{\min}}$, no *Passo* 3. Com essa estratégia, o algoritmo inicialmente procura resolver a infactibilidade de metas invertendo o mínimo possível de metas, e com a persistência na infactibilidade a ênfase se desloca para liberar mais e mais variáveis de suas metas não atendidas.

Suponha uma instância de programação inteira 0–1 com $m = 100$ restrições e $n = 1000$ variáveis binárias. Um exemplo da evolução do controle adaptativo das cardinalidades dos conjuntos D_0, G_p e G_s é exibido na Tabela 2.1 considerando $\delta_1 = 0,5$, $\delta_2 = 1,05$, $f_{d_{\max}} = 0,25$, $k_4 = 0,5$, $g_{p_{\min}} = 1$, $g_{s_{\min}} = 0$, $g_{\max} = 2$ e $f_s = 0,1$. Para facilitar a explicação a seguir, assume-se que $|D| = 100$ em qualquer PL' com infactibilidade inteira, ou seja, nunca ocorre degenerescência de bases nessa

situação. Como não há variáveis básicas degeneradas na infactibilidade inteira, todas as variáveis básicas assumem valores fracionários. Além disso, outra simplificação didática foi adotar $|G| = 50$ ao longo de todos os PL's com infactibilidade de metas, fornecendo $f_s \times |G| = 5$. Note que restrições tabu excluem variáveis com metas não atendidas do conjunto G.

Nas iterações de 1 a 20, a BTP mantém-se na infactibilidade inteira. Como $f_d = 0,25$ e $|D| = 100$, por (2.21), $f_d \times |D| = 0{,}25 \times 100$ e ao final da vigésima iteração o PL' contará com 500 metas estabelecidas. Ao transitar para a infactibilidade de metas na iteração 21, como $\delta_1 = 0,5$, f_d é atualizado em 0,125. A infactibilidade de metas persiste até a iteração 221. A cada $k_4 \times m = 50$ iterações, g_s e g_p são alternadamente incrementados em uma unidade até atingirem $g_{\max} = 2$. Na iteração 221, transcorridas 50 iterações com $g_p = g_s = g_{\max}$, a ênfase transita para a liberação de metas com $g_p = 1$ e $g_s = f_s \times |G| = 5$. A infactibilidade de metas é resolvida nessa mesma iteração. Nas 50 iterações seguintes, que vão de 222 a 271 novas metas, $f_d \times |D| = 0{,}125 \times 100 = 13$ por iteração, são estabelecidas para PL's com infactibilidade inteira. Essa permanência continuada na infactibilidade inteira por $k_4 \times m = 50$ iterações, sugere um aumento em f_d ao multiplicá-lo por $\delta_2 = 1,05$ na iteração 272.

Tab. 2.1: Exemplo de controle adaptativo em D_0, G_p e G_s.

Iteração	Estado	f_d	g_p	g_s	Metas
inicio	Infact. inteira	0,25	1	0	0
1 a 20	Infact. Inteira	0,25	1	0	500
21	Infact. de metas	0,125	1	0	500
22 a 70	Infact. de metas	0,125	1	0	500
71 a 120	Infact. de metas	0,125	1	1	450
121 a 170	Infact. de metas	0,125	2	1	400
171 a 220	Infact. de metas	0,125	2	2	300
221	Infact. de metas	0,125	1	5	295
222	Infact. inteira	0,125	1	0	308
223 a 271	Infact. Inteira	0,125	1	0	945
272	Infact. Inteira	0,13125	1	0	959

2.3.14 Integração entre as componentes da BTP

A Figura 2.5 mostra um fluxograma simplificado da BTP considerando-se que o PL original é resolvido antes do início da busca. Em cada início de iteração, é realizada a reotimização do PL'. Quando uma solução incumbente é encontrada, o PL' pode tornar-se infactível após a atualização da restrição de função objetivo. Se isso ocorrer supõe-se que a solução encontrada é ótima com o término

do algoritmo, desde, é claro, que o ε da restrição de função objetivo seja suficientemente pequeno. Se a solução corrente do PL' não é inteira, então há duas possibilidades. O problema é infactível nas metas, há variáveis fracionárias com metas não atendidas ou todas as metas estabelecidas são atendidas, mas restam variáveis fracionárias, caracterizando a infactibilidade inteira. Note que os critérios tabu e de aspiração são considerados apenas na infactibilidade de metas. Sob infactibilidade inteira, variáveis fracionárias que ainda não possuem metas e que fazem parte do conjunto D_0 recebem metas, então o conjunto N' tem sua cardinalidade aumentada em $d_0 = |D_0|$. Sob a infactibilidade de metas, as metas das variáveis do conjunto G_p são invertidas e as variáveis do conjunto G_s são liberadas de suas metas, diminuindo assim a cardinalidade do conjunto N' em $g_s = |G_s|$. As novas metas surgidas da infactibilidade inteira ou da infactibilidade de metas permitem construir um novo PL'. Se o critério de parada por tempo computacional for satisfeito, a busca termina. Caso contrário, a busca inicia uma nova iteração.

2.4 Busca tabu paramétrica passo a passo

Seja a instância de mochilas múltiplas PET1 (Petersen 1967) representada pelo PL (2.24) contendo 6 variáveis, 10 restrições estruturais mais uma restrição de função objetivo na linha 1. O valor da função objetivo para a solução ótima é -3800. No início, a restrição de função objetivo aparece relaxada com seu lado direito em 100.000.000. Como o problema é de maximização da utilidade e a implementação de BTP trabalha apenas com minimização, os coeficientes aparecem com sinais negativos na função objetivo. Para facilitar o entendimento, adotou-se $|D_0| = 1$, $|G_p| = 1$, $|G_s| = 1$ em todas as iterações do procedimento. Para medir a infactilibidade inteira e a resistência a metas, a distância à meta DM foi utilizada e a escolha preferencial é calculada com (2.10) assumindo $w = 0, 1$. A aspiração por resistência não foi considerada para esse exemplo. A abordagem simples de $M_j = 1$ para todo $j \in \{1, \ldots, n\}$ foi adotada na fase I do PL' ao impor $r = 0$ em (2.7). Além disso, $T^0 = 1$.

A Tabela 2.2 apresenta a solução ótima do PL com $N' = \{\}$, ou seja, quando não há nenhuma meta estabelecida. A linha 1 contém os nomes das variáveis, $x_1, \ldots, x_6$. A linha 2 indica as metas das variáveis e como a busca acabou de iniciar, todas encontram-se livres. A linha 3 mostra o resultado da fase II do PL de duas fases. As variáveis de interesse são x_4 e x_5 por assumirem valores fracionários. A linha 4 indica até qual iteração uma restrição tabu existe para cada variável. A presença do texto "NÃO" indica ausência de restrição. A busca encontra-se na infactibilidade inteira porque não há nenhuma meta violada e por haver variáveis inteiras que assumem valores fracionários. Assim, as linhas 5, 6 e 7 representam respectivamente as penalidades inteiras *UP* e *DN* e a medida de escolha preferencial. As células dessas linhas para as variáveis que não são fracionárias ficam vazias. A linha 8 representa a pertinência das variáveis ao conjunto D, e então $D = \{4, 5\}$. A linha 9 é relativa ao conjunto D_0, e assim deve conter as variáveis com maiores valores de escolha preferencial. Como

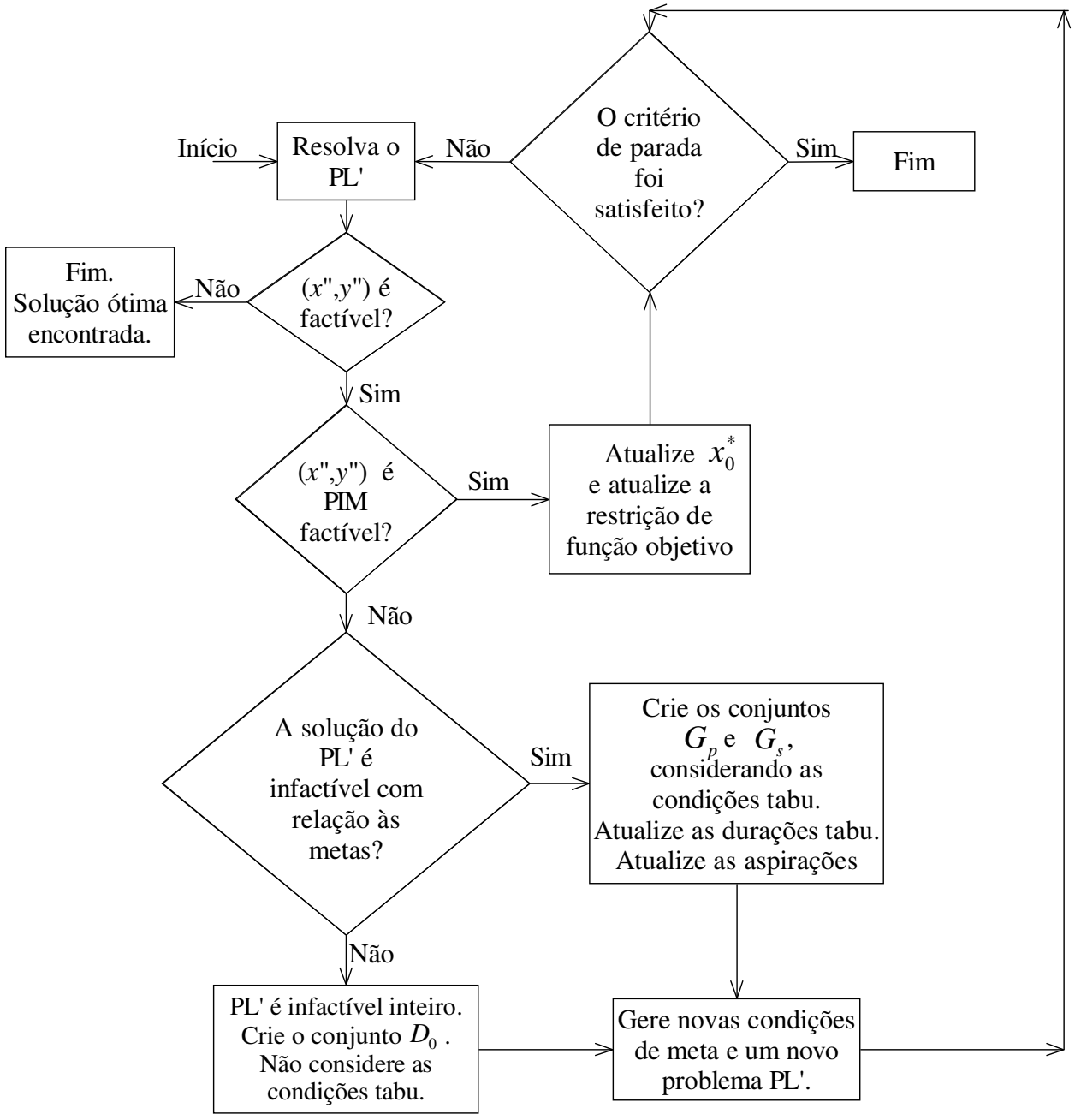

Fig. 2.5: Busca tabu paramétrica.

foi assumido $|D_0| = 1$, $D_0 = \{5\}$. A linha 10 exibe se respostas *DN* ou *UP* devem ser aplicadas de acordo com as expressões (2.13) e (2.14). Como $IP_5(DN) < IP_5(UP)$, a expressão (2.13) é usada e uma transição *T-DN* é aplicada à variável x_5. A linha 11 é relativa ao valor da função objetivo e a linha 12 mostra a iteração atual.

A partir do conjunto D_0, a construção do PL' da iteração 1 considera $N^- = \{5\}$ e $N^+ = \{\}$. Como foi assumido $M_j = 1$ para todas as variáveis com metas, a função objetivo da fase I da iteração 1 é $z = \min +1x_5$. Ao final da fase I da iteração 1, a variável x_5 atende a sua meta, ou seja, assume o valor zero e é fixada nesse valor no PL da fase II. A variável x_4, por ser a única fracionária, compõe os conjuntos D e D_0 conforme a Tabela 2.3. Com a nova meta *DN* estabelecida em x_4, $N^- = \{4, 5\}$, $N^+ = \{\}$ e a função objetivo da fase I da iteração 2 é $z = \min +1x_4 + 1x_5$. Ao final da iteração 2 (Tabela 2.4), a variáveis x_1 fica fracionária e pela expressão (2.14) recebe uma meta *UP*. Assim, $N^- = \{4, 5\}$, $N^+ = \{1\}$, e a função objetivo da fase I da iteração 3 é $z = \min -1x_1 + 1x_4 + 1x_5$. No final das iterações 3 (Tabela 2.5), 4 (Tabela 2.6) e 5 (Tabela 2.7), as variáveis fracionárias são respectivamente x_2, x_6 e x_3. No final da iteração 5, $N^- = \{4, 5\}$ e $N^+ = \{1, 2, 3, 6\}$. Assim, a função objetivo da fase I da iteração 6 é $z = \min -1x_1 - 1x_2 - 1x_3 + 1x_4 + 1x_5 - 1x_6$.

Na iteração 6, o PL torna-se infactível nas metas com a informação pertinente na Tabela 2.8. As linhas 1 e 2 trazem os nomes das variáveis e as metas. A linha 3 exibe os custos reduzidos das variáveis na solução ótima da fase I do PL. A linha 4 traz a solução ótima da fase II do PL e a linha 5 as restrições tabu. As linhas 6, 7 são relativas às resistências diretas a metas *UP* e *DN*. As linhas 8 e 9 são relativas às resistências potenciais a metas *UP* e *DN*. A linha 10 representa as variáveis com maiores valores de resistência que formam o conjunto G. As linhas 11 e 12 exibem as composições dos conjuntos G_p e G_s respectivamente. A linha 13 indica as novas metas, a linha 14 o valor da função objetivo e a linha 15 a iteração atual.

Ao final da fase I da iteração 6, a variável x_6 não atende a meta *UP* estabelecida e fica fracionária. Sendo assim, na fase II não pode ser fixada em 1, assume um valor fracionário, é considerada diretamente infactível com relação a sua meta e faz parte do conjunto G. Além disso, como $T^0 = 1$, a variável x_2 é escolhida como potencialmente infactível em relação a sua meta por apresentar o maior valor de GR_j^0. No ordenamento de resistência a metas, as diretamente infactíveis são prioritárias. Então x_6 faz parte de G_p e terá sua meta invertida, enquanto que x_2 faz parte de G_s e será liberada de sua meta de acordo com a Tabela 2.8. Caso x_6 fosse tabu, x_2 entraria em G_p e sua meta seria invertida. Esse tratamento leva à descoberta de uma solução inteira factível com valor -1900 na iteração 7a apresentada na Tabela 2.9. Ainda nessa iteração, a restrição de função objetivo é atualizada com $\varepsilon = 1$, tornando essa solução infactível. O PL é reotimizado e na Tabela 2.10 x_4 e x_5 são, respectivamente, diretamente e potencialmente infactíveis em relação a suas metas. Note que x_6 apresenta custo reduzido inferior ao de x_5, mas não pode ser classificada como potencialmente infactível em

relação a sua meta devido a uma restrição tabu que permanecerá até a iteração 8.

$$
\begin{aligned}
& x_0 = \min -100x_1 - 600x_2 - 1200x_3 - 2400x_4 - 500x_5 - 2000x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcc}
-100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \\
8x_1 & +12x_2 & +13x_3 & +64x_4 & +22x_5 & +41x_6 & \leq & 80 \\
8x_1 & +12x_2 & +13x_3 & +75x_4 & +22x_5 & +41x_6 & \leq & 96 \\
3x_1 & +6x_2 & +4x_3 & +18x_4 & +6x_5 & +4x_6 & \leq & 20 \\
5x_1 & +10x_2 & +8x_3 & +32x_4 & +6x_5 & +12x_6 & \leq & 36 \\
5x_1 & +13x_2 & +8x_3 & +42x_4 & +6x_5 & +20x_6 & \leq & 44 \\
5x_1 & +13x_2 & +8x_3 & +48x_4 & +6x_5 & +20x_6 & \leq & 48 \\
 & & & & +8x_5 & & \leq & 10 \\
3x_1 & & +4x_3 & & +8x_5 & & \leq & 18 \\
3x_1 & +2x_2 & +4x_3 & & +8x_5 & +4x_6 & \leq & 22 \\
3x_1 & +2x_2 & +4x_3 & +8x_4 & +8x_5 & +4x_6 & \leq & 24
\end{array} \\
& 0 \leq x_j \leq 1, \quad j \in \{1, \ldots, 6\}
\end{aligned}
\tag{2.24}
$$

Tab. 2.2: Exemplo da BTP - Solução do PL original.

Variáveis	x_1	x_2	x_3	x_4	x_5	x_6
Metas	*Livre*	*Livre*	*Livre*	*Livre*	*Livre*	*Livre*
Fase II	0,000	0,000	1,000	0,363	0,126	1,000
Tabu	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO
$IP_j(UP)$				0,637	0,874	
$IP_j(DN)$				0,363	0,126	
CP_j				0,284	0,758	
D	NÃO	NÃO	NÃO	SIM	SIM	NÃO
D_0	NÃO	NÃO	NÃO	NÃO	SIM	NÃO
Novas metas					*T-DN*	
Objetivo	-4134,074					
Iteração	0					

Iteração 1 - Fase I

$$
\begin{aligned}
& z = \min +1x_5 \\
& \text{sa} \\
& \quad -100x_1 \quad -600x_2 \quad -1200x_3 \quad -2400x_4 \quad -500x_5 \quad -2000x_6 \quad \leq \quad 100000000 \\
& \vdots \\
& 0 \leq x_j \leq 1, \quad j \in \{1, \ldots, 6\}
\end{aligned}
$$

Iteração 1 - Fase II

$$
\begin{aligned}
& x_0 = \min -100x_1 - 600x_2 - 1200x_3 - 2400x_4 - 500x_5 - 2000x_6 \\
& \text{sa} \\
& \quad -100x_1 \quad -600x_2 \quad -1200x_3 \quad -2400x_4 \quad -500x_5 \quad -2000x_6 \quad \leq \quad 100000000 \\
& \vdots \\
& 0 \leq x_5 \leq 0 \\
& 0 \leq x_j \leq 1, \quad j \in \{1, 2, 3, 4, 6\}
\end{aligned}
$$

Tab. 2.3: Exemplo da BTP - Iteração 1.

Variáveis	x_1	x_2	x_3	x_4	x_5	x_6
Metas	*Livre*	*Livre*	*Livre*	*Livre*	*DN*	*Livre*
Fase II	0,000	0,000	1,000	0,381	0,000	1,000
Tabu	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO
$IP_j(UP)$				0,619		
$IP_j(DN)$				0,381		
CP_j				0,248		
D	NÃO	NÃO	NÃO	SIM	NÃO	NÃO
D_0	NÃO	NÃO	NÃO	SIM	NÃO	NÃO
Novas metas				*T-DN*		
Objetivo	-4114,286					
Iteração	1					

Iteração 2 - Fase I

$$
\begin{aligned}
& z = \min +1x_4 + 1x_5 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 0 \leq x_j \leq 1, \quad j \in \{1, \ldots, 6\}
\end{aligned}
$$

Iteração 2 - Fase II

$$
\begin{aligned}
& x_0 = \min -100x_1 - 600x_2 - 1200x_3 - 2400x_4 - 500x_5 - 2000x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 0 \leq x_4 \leq 0, 0 \leq x_5 \leq 0 \\
& 0 \leq x_j \leq 1, \quad j \in \{1, 2, 3, 6\}
\end{aligned}
$$

Tab. 2.4: Exemplo da BTP - Iteração 2.

Variáveis	x_1	x_2	x_3	x_4	x_5	x_6
Metas	*Livre*	*Livre*	*Livre*	*DN*	*DN*	*Livre*
Fase II	0,600	1,000	1,000	0,000	0,000	1,000
Tabu	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO
$IP_j(UP)$	0,400					
$IP_j(DN)$	0,600					
CP_j	0,210					
D	SIM	NÃO	NÃO	NÃO	NÃO	NÃO
D_0	SIM	NÃO	NÃO	NÃO	NÃO	NÃO
Novas metas	*T-UP*					
Objetivo	-3860,000					
Iteração	2					

Iteração 3 - Fase I

$$
\begin{aligned}
& z = \min -1x_1 + 1x_4 + 1x_5 \\
& \text{sa} \\
& \begin{array}{rrrrrrcr} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 0 \leq x_j \leq 1, \quad j \in \{1, \ldots, 6\}
\end{aligned}
$$

Iteração 3 - Fase II

$$
\begin{aligned}
& x_0 = \min -100x_1 - 600x_2 - 1200x_3 - 2400x_4 - 500x_5 - 2000x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcr} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 1 \leq x_1 \leq 1, 0 \leq x_4 \leq 0, 0 \leq x_5 \leq 0 \\
& 0 \leq x_j \leq 1, \quad j \in \{2, 3, 6\}
\end{aligned}
$$

Tab. 2.5: Exemplo da BTP - Iteração 3.

Variáveis	x_1	x_2	x_3	x_4	x_5	x_6
Metas	*UP*	*Livre*	*Livre*	*DN*	*DN*	*Livre*
Fase II	1,000	0,846	1,000	0,000	0,000	1,000
Tabu	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO
$IP_j(UP)$		0,154				
$IP_j(DN)$		0,846				
CP_j		0,702				
D	NÃO	SIM	NÃO	NÃO	NÃO	NÃO
D_0	NÃO	SIM	NÃO	NÃO	NÃO	NÃO
Novas metas		*T-UP*				
Objetivo	-3807,692					
Iteração	3					

Iteração 4 - Fase I

$$
\begin{aligned}
& z = \min -1x_1 - 1x_2 + 1x_4 + 1x_5 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 0 \leq x_j \leq 1, \quad j \in \{1, \ldots, 6\}
\end{aligned}
$$

Iteração 4 - Fase II

$$
\begin{aligned}
& x_0 = \min -100x_1 - 600x_2 - 1200x_3 - 2400x_4 - 500x_5 - 2000x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 1 \leq x_1 \leq 1, 1 \leq x_2 \leq 1, 0 \leq x_4 \leq 0, 0 \leq x_5 \leq 0 \\
& 0 \leq x_j \leq 1, \quad j \in \{3, 6\}
\end{aligned}
$$

Tab. 2.6: Exemplo da BTP - Iteração 4.

Variáveis	x_1	x_2	x_3	x_4	x_5	x_6
Metas	*UP*	*UP*	*Livre*	*DN*	*DN*	*Livre*
Fase II	1,000	1,000	1,000	0,000	0,000	0,900
Tabu	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO
$IP_j(UP)$						0,100
$IP_j(DN)$						0,900
CP_j						0,810
D	NÃO	NÃO	NÃO	NÃO	NÃO	SIM
D_0	NÃO	NÃO	NÃO	NÃO	NÃO	SIM
Novas metas						*T-UP*
Objetivo	-3700,000					
Iteração	4					

Iteração 5 - Fase I

$$
\begin{aligned}
& z = \min -1x_1 - 1x_2 + 1x_4 + 1x_5 - 1x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 0 \leq x_j \leq 1, \quad j \in \{1, \ldots, 6\}
\end{aligned}
$$

Iteração 5 - Fase II

$$
\begin{aligned}
& x_0 = \min -100x_1 - 600x_2 - 1200x_3 - 2400x_4 - 500x_5 - 2000x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 1 \leq x_1 \leq 1, 1 \leq x_2 \leq 1, , 0 \leq x_4 \leq 0, 0 \leq x_5 \leq 0, 1 \leq x_6 \leq 1 \\
& 0 \leq x_j \leq 1, \quad j \in \{3\}
\end{aligned}
$$

Tab. 2.7: Exemplo da BTP - Iteração 5.

Variáveis	x_1	x_2	x_3	x_4	x_5	x_6
Metas	*UP*	*UP*	*Livre*	*DN*	*DN*	*UP*
Fase II	1,000	1,000	0,750	0,000	0,000	1,000
Tabu	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO
$IP_j(UP)$			0,250			
$IP_j(DN)$			0,750			
CP_j			0,510			
D	NÃO	NÃO	SIM	NÃO	NÃO	NÃO
D_0	NÃO	NÃO	SIM	NÃO	NÃO	NÃO
Novas metas			*T-UP*			
Objetivo	-3600,000					
Iteração	5					

Iteração 6 - Fase I

$$
\begin{aligned}
& z = \min -1x_1 - 1x_2 - 1x_3 + 1x_4 + 1x_5 - 1x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 0 \leq x_j \leq 1, \quad j \in \{1, \ldots, 6\}
\end{aligned}
$$

Iteração 6 - Fase II

$$
\begin{aligned}
& x_0 = \min -100x_1 - 600x_2 - 1200x_3 - 2400x_4 - 500x_5 - 2000x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 1 \leq x_1 \leq 1, 1 \leq x_2 \leq 1, 1 \leq x_3 \leq 1, 0 \leq x_4 \leq 0, 0 \leq x_5 \leq 0 \\
& 0 \leq x_j \leq 1, \quad j \in \{6\}
\end{aligned}
$$

Tab. 2.8: Exemplo da BTP - Iteração 6.

Variáveis	x_1	x_2	x_3	x_4	x_5	x_6
Metas	*UP*	*UP*	*UP*	*DN*	*DN*	*UP*
CR	-0,750	-0,350	-0,600	3,100	1,300	0,000
Fase II	1,000	1,000	1,000	0,000	0,000	0,900
Tabu	NÃO	NÃO	NÃO	NÃO	NÃO	NÃO
$GR_j(UP)$						0,100
$GR_j(DN)$						
$GR_j^0(UP)$	-0,750	-0,350	-0,600			
$GR_j^0(DN)$				-3,100	-1,300	
G	NÃO	SIM	NÃO	NÃO	NÃO	SIM
G_p	NÃO	NÃO	NÃO	NÃO	NÃO	SIM
G_s	NÃO	SIM	NÃO	NÃO	NÃO	NÃO
Novas metas		*T-Livre*				*T-DN*
Objetivo	-3700,000					
Iteração	6					

Iteração 7a - Fase I

$$
\begin{aligned}
& z = \min -1x_1 - 1x_3 + 1x_4 + 1x_5 + 1x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 0 \leq x_j \leq 1, \quad j \in \{1, \ldots, 6\}
\end{aligned}
$$

Iteração 7a - Fase II

$$
\begin{aligned}
& x_0 = \min -100x_1 - 600x_2 - 1200x_3 - 2400x_4 - 500x_5 - 2000x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & 100000000 \end{array} \\
& \vdots \\
& 1 \leq x_1 \leq 1, 1 \leq x_3 \leq 1, 0 \leq x_4 \leq 0, 0 \leq x_5 \leq 0, 0 \leq x_6 \leq 0 \\
& 0 \leq x_j \leq 1, \quad j \in \{2\}
\end{aligned}
$$

Tab. 2.9: Exemplo da BTP - Iteração 7a, solução encontrada.

Variáveis	x_1	x_2	x_3	x_4	x_5	x_6
Metas	*UP*	*Livre*	*UP*	*DN*	*DN*	*DN*
Fase II	1,000	1,000	1,000	0,000	0,000	0,000
Tabu	NÃO	NÃO	NÃO	NÃO	NÃO	8
Objetivo	-1900,000					
Iteração	7					

Iteração 7b - Fase I

$$
\begin{aligned}
& \min z = -1x_1 - 1x_3 + 1x_4 + 1x_5 + 1x_6 \\
& \text{sa} \\
& \begin{array}{rrrrrrcl} -100x_1 & -600x_2 & -1200x_3 & -2400x_4 & -500x_5 & -2000x_6 & \leq & -1901 \end{array} \\
& \vdots \\
& 0 \leq x_j \leq 1, \quad j \in \{1, \ldots, 6\}
\end{aligned}
$$

Iteração 7b - Fase II

$$
\begin{aligned}
& x_0 = \min -100x_1 - 600x_2 - 1200x_3 - 2400x_4 - 500x_5 - 2000x_6 \\
& \text{sa} \\
& \quad -100x_1 \quad -600x_2 \quad -1200x_3 \quad -2400x_4 \quad -500x_5 \quad -2000x_6 \quad \leq \quad -1901 \\
& \vdots \\
& 1 \leq x_1 \leq 1, 1 \leq x_3 \leq 1, , 0 \leq x_5 \leq 0, 0 \leq x_6 \leq 0 \\
& 0 \leq x_j \leq 1, \quad j \in \{2, 4\}
\end{aligned}
$$

Tab. 2.10: Exemplo da BTP - Iteração 7b.

Variáveis	x_1	x_2	x_3	x_4	x_5	x_6
Metas	*UP*	*Livre*	*UP*	*DN*	*DN*	*DN*
CR	-1,042	-0,250	-1,500	0,000	0,792	0,167
Fase II	1,000	0,000	1,000	0,719	0,000	0,000
Tabu	NÃO	NÃO	NÃO	NÃO	NÃO	8
$GR_j(UP)$						
$GR_j(DN)$				0,719		
$GR_j^0(UP)$	-1,042		-1,500			
$GR_j^0(DN)$					-0,792	-0,167
G	NÃO	SIM	NÃO	SIM	SIM	NÃO
G_p	NÃO	NÃO	NÃO	SIM	NÃO	NÃO
G_s	NÃO	NÃO	NÃO	NÃO	SIM	NÃO
Novas metas				*T-UP*	*T-Livre*	
Objetivo	-3025,000					
Iteração	7					

2.4.1 Um paralelo entre BTP e *branch-and-bound*

Como a busca tabu paramétrica surgiu de uma crítica lançada à rigidez da memória estática da busca em árvore, é oportuno correlacionar as componentes internas existentes em cada uma das duas abordagens. Ambas abordagens são baseadas na relaxação linear do PIM. A Tabela 2.11 resume as técnicas empregadas para solucionar obstáculos comuns, como o uso de memória, o tratamento da infactibilidade em um PL, a infactibilidade inteira, quando variáveis inteiras assumem valores fracionários e como cada uma das abordagens evita a exploração de regiões com soluções de pior qualidade que a incumbente.

Tab. 2.11: Comparativo das abordagens da BTP com o *branch-and-bound*.

	busca tabu paramétrica	*branch-and-bound*
Memória	Usa memória tabu flexível. As decisões são reavaliadas já que a memória tabu tem prazo de validade.	As ramificações são rígidas impondo uma dicotomia nas variáveis. Uma vez imposta não é mais reavalida.
Infactibilidade no PL	É detectada quando metas estabelecidas não podem ser atendidas simultaneamente, e o PL' é classificado como infactível nas metas. O conjunto G é identificado bem como os seus subconjuntos G_p e G_s. Assim, a maneira da BTP resolver a infactibilidade do PL' se dá pela inversão de metas não atendidas e adição de restrições tabu para variáveis em G_p. Já as variáveis em G_s são liberadas de suas metas.	Faz *backtrack* na árvore de enumeração. As subárvores do nó infactível não precisam ser avaliadas.
Infactibilidade Inteira	Identificação do conjunto D das variáveis com infactibilidade inteira e imposição de novas metas a variáveis do subconjunto D_0. Lembrando que a infactibilidade inteira se dá com variáveis binárias que assumem valores fracionários e não têm metas associadas.	Criação de dois ramos em uma única variável, produzindo dois novos nós.
Controle da qualidade das soluções	Uso da restrição de função objetivo que controla a qualidade de acordo com o valor da função objetivo da solução incumbente.	Faz poda. Se um nó apresenta um limitante pior que o da solução incumbente, ele pode ser descartado.
Garantia de otimalidade	Não.	Sim.
Ajuste de parâmetros	Alto.	Baixo.

Capítulo 3

Extensões e Uso de Memória de Longo Prazo

Na primeira seção deste capítulo, são apresentados alguns procedimentos que podem ser acoplados à BTP como componentes coadjuvantes ou até mesmo como abordagens alternativas a algumas técnicas apresentadas no Capítulo 2. Na segunda seção são desenvolvidos os métodos fundamentados em conjunto referência de soluções, compondo assim os módulos baseados em memória de longo prazo e que complementam o algoritmo básico da busca tabu paramétrica.

3.1 Extensões de curto prazo

A BTP conforme apresentada no Capítulo 2 pode receber algumas técnicas complementares que ainda mantêm o caráter de curto prazo em relação ao uso de memória. As técnicas abordadas envolvem o uso de *probing* que por implicações lógicas cria metas *UP* ou *DN*, atualização do lado direito da restrição de função objetivo com o intuito de encontrar mais soluções inteiras-factíveis ao longo da busca, fixação de variáveis pela análise de custo reduzido usando a relaxação linear do PL original como referência, uso de cortes gerados na fase I do PL', que podem complementar ou substituir a memória tabu, e o uso de *branch-and-cut* como resolvedor de infactibilidade de metas intratáveis. O uso de cortes é sugerido por Glover 2006 e as demais técnicas são propostas do autor deste trabalho.

3.1.1 Técnicas de *probing*

As técnicas de *probing* (Savelsbergh, 1994) fixam variáveis binárias em zero ou um por implicações lógicas e podem ser aplicadas em qualquer nó do *branch-and-cut*. Por exemplo, se a tentativa de fixar uma variável em zero leva a uma infactibilidade, então essa variável só pode assumir o valor um e vice-versa. Quatro regras foram usadas e o detalhamento está descrito no Apêndice A. A implementação inicial do algoritmo, restrita a programação inteira 0–1 pura, apresentou pouco im-

pacto, e então a implementação em programação inteira mista foi descartada. Na BTP, ao contrário de fixar as variáveis como ocorre no *branch-and-cut*, os testes de *probing* estabelecem metas *UP* ou *DN*. Algumas das metas estabelecidas pelos testes de *probing* numa iteração podem entrar em conflito com as estabelecidas parametricamente pela BTP e decisões precisam ser tomadas nas seguintes circunstâncias:

1. Quando o algoritmo de *probing* sugere na iteração *Iter* metas contrárias às estabelecidas em iterações anteriores a *Iter* pode-se:
   (a) Cancelar completamente o tratamento de *probing* na iteração *Iter*.
   (b) Aplicar as metas de *probing* invertendo metas estabelecidas em outras iterações, além, é claro, das metas que não apresentam conflitos.
   (c) Não aplicar as metas de *probing* que entram em conflito, mas aplicar as que não apresentam conflito.
2. Quando o algoritmo de *probing* sugere metas numa iteração *Iter* e nessa mesma iteração *Iter* a BTP sugere metas conflitantes com as do *probing* pode-se:
   (a) Aplicar as metas estabelecidas pelo *probing*.
   (b) Aplicar as metas estabelecidas parametricamente.

Todas as combinações foram testadas, e a melhor configuração é composta das regras 1(a) e 2(a).

3.1.2 Atualização no lado direito da restrição de função objetivo

A restrição de função objetivo $cx+dy \leq x_0^* - \varepsilon$, em que x_0^* é o valor da melhor solução conhecida do PIM e ε é um número positivo pequeno, é atualizada sempre que uma solução inteira-factível é encontrada. Quanto menor for o valor x_0^*, mais essa restrição terá impacto na solução do PL' e maior será a sua influência nos cálculos de penalidade inteira e resistência a metas. Uma maneira de controlar essa influência consiste em adicionar uma relaxação $relax$ positiva no lado direito da restrição, no intuito de promover um tipo de diversificação ao provocar maior variabilidade nos cálculos de $IP_j(UP)$, $IP_j(DN)$, $GR_j(UP)$ e $GR_j(DN)$. A restrição toma então a forma $cx + dy \leq x_0^* - \varepsilon + |x_0^*| \times relax$ em que $relax$ é atualizada segundo a expressão (3.1).

$$relax = \begin{cases} 0 & \text{se} \quad Iter < UltIterSol + \Delta ss \\ \frac{MaxRelax}{\Delta rm}\left(Iter - (UltIterSol + \Delta ss)\right) & \text{se} \quad UltIterSol + \Delta ss \leq Iter \leq UltIterSol + \Delta ss + \Delta rm \\ MaxRelax & \text{se} \quad Iter > UltIterSol + \Delta ss + \Delta rm \end{cases} \tag{3.1}$$

Em (3.1), *Iter* é a iteração corrente, *UltIterSol* é a última iteração em que uma solução inteira-factível foi encontrada. Transcorridas Δss iterações sem encontrar solução após *UltIterSol*, a relaxação começa a ser aplicada e aumenta linearmente com o passar das iterações até atingir o máximo *MaxRelax*. A representação gráfica da expressão (3.1) na Figura 3.1 mostra a inclinação α do segmento de reta $\frac{MaxRelax}{\Delta rm}$, em que Δrm é o número de iterações em que a relaxação aumenta linearmente até estabilizar em $MaxRelax$.

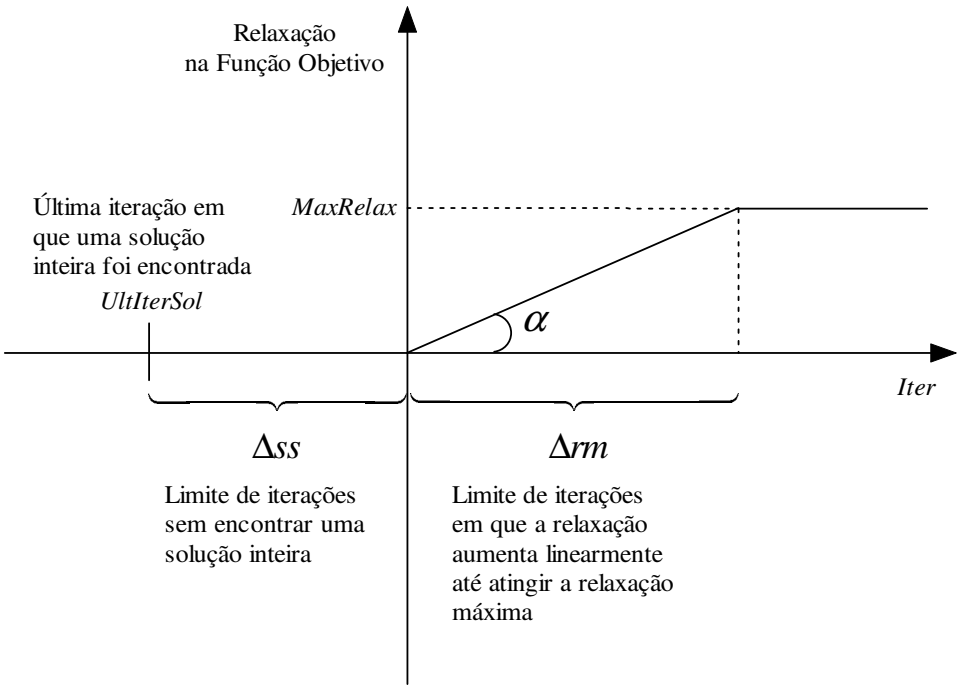

Fig. 3.1: Relaxação aplicada à restrição de função objetivo.

3.1.3 Fixação de variáveis por análise de custo reduzido

A fixação de variáveis pela análise de custo reduzido é usada em algoritmos de *branch-and-cut* (Atamtürk e Savelsbergh, 2005) e pela sua simplicidade pode ser facilmente integrada ao algoritmo da BTP. Para aplicar esse teste, é preciso guardar o valor da função objetivo da relaxação linear do PL original, x_0^{PL}, bem como seu vetor dos custos reduzidos das variáveis não-básicas. A função objetivo do modelo pode ser escrita no espaço reduzido das variáveis não-básicas como $x_0 = x_0^{PL} + \sum_{j \in R'} CR_j x_j + \sum_{j \in R''} CR_j y_j$, sabendo que $R = R' \cup R''$ é o conjunto das variáveis não-básicas e CR_j é o custo reduzido de uma variável. Seja então x_0^* o valor da função objetivo da melhor solução inteira-factível conhecida. As variáveis x_j com $j \in R'$ são do tipo 0–1 e estão canalizadas, $0 \leq x_j \leq 1$.

Como são não-básicas, assumem apenas os valores 0 ou 1. Sem perda de generalidade, podemos assumir que na solução ótima do PL original, todas estão em 0 desde que as variáveis em 1 sejam trocadas pelas suas variáveis de folga, $s_j = 1 - x_j$. Seja uma variável $j \in R'$ que assume o valor zero tanto na solução ótima do PL original quanto na solução incumbente. Se essa variável for forçada a assumir o valor 1, provocará uma degradação de valor $1 \times CR_j$ na função objetivo do PL original. Assim, se a degradação no PL resultar em um valor de função objetivo que supera o valor da solução incumbente, essa variável pode ser fixada, ou seja, se $x_0^{PL} + CR_j \geq x_0^*$, então $x_j = 0$. Atamtürk e Savelsbergh, 2005 advertem que esse tipo de fixação é eficiente quando o *gap* entre a solução inteira ótima e a solução do PL é pequeno e quando há grandes variações nos coeficientes da função objetivo.

3.1.4 Uso de cortes gerados na fase I do PL'

A busca tabu paramétrica produz cortes válidos que podem ser usados para complementar ou até mesmo substituir a memória tabu apresentada na Subseção 2.3.9. A abordagem de duas fases para resolver o PL' é muito conveniente para a descrição desses cortes. Quando o valor M_j inteiro é usado na fase I, a função objetivo torna-se $z_0 = \min \sum_{j \in N^-} M_j x_j + \sum_{j \in N^+} M_j (1 - x_j)$ e que resulta em um valor $z_0 = z_0''$ associado a uma solução $x = x''$ do PL'. Pode-se forçar então $z_0 \geq \lceil z_0'' \rceil$ produzindo a inequação 3.2.

$$\sum_{j \in N^-} M_j x_j + \sum_{j \in N^+} M_j (1 - x_j) \geq \lceil z_0'' \rceil \tag{3.2}$$

A inequação 3.2 é válida para todas as soluções inteiras-factíveis do PIM e se for gerada a partir de uma solução infactível nas metas, exclui a solução $x = x''$ que produziu essa infactibilidade de metas. Por outro lado, se todas as metas são atendidas, $x' = x''$, mas nenhuma solução inteira-factível é encontrada (infactibilidade inteira), o valor z_0'' vale zero e nenhum corte pode ser derivado. E finalmente, quando a solução do PL' resulta em uma solução inteira-factível, z_0'' é um valor inteiro e então o corte pode ser modificado gerando a inequação 3.3.

$$\sum_{j \in N^-} M_j x_j + \sum_{j \in N^+} M_j (1 - x_j) \geq z_0'' + 1 \tag{3.3}$$

No *branch-and-bound* paramétrico (Glover, 1978), ramificações que deixam de exercer influência são removidas por serem supérfluas, ou seja, a remoção não altera a solução ótima do nó corrente. A mesma idéia pode ser aplicada à BTP, lembrando que a função objetivo da fase I do PL' pode ser escrita no espaço das variáveis não-básicas como $z_0 = z_0'' + \sum_{j \in R} CR_j x_j$, em que R é o conjunto das variáveis não-básicas e CR_j é o custo reduzido de uma variável. As variáveis x_j com $j \in R$ são do

tipo 0–1 e estão canalizadas, $0 \leq x_j \leq 1$. Como são não-básicas, assumem apenas os valores 0 ou 1. Por simplicidade podemos assumir que todas estão em 0 desde que as variáveis em 1 sejam trocadas pelas suas variáveis de folga, $s_j = 1 - x_j$. Se uma variável x_j apresentar $CR_j > M_j$, então ainda que o valor do seu coeficiente na função objetivo seja reduzido em M_j, o custo reduzido dessa variável continuará maior que zero e essa variável continuará não-básica no seu valor atual, ou seja, $x_j = 0$ ou $x_j = 1$. Sendo assim, a meta estabelecida nessa variável é supérflua e as inequações (3.2) e (3.3) podem ser apertadas ao remover as variáveis com custo reduzido maior que M_j.

O número máximo de cortes permitidos é controlado pelo parâmetro **MaxCortes**, os cortes são excluídos quando não estão mais ativos, mas isso só pode acontecer após **MinDuracaoCortes** iterações após o corte ter sido imposto.

3.1.5 Uso de *branch-and-cut* para resolver infactibilidade de metas

Às vezes, para um certo conjunto de variáveis binárias há um número reduzido de possibilidades de atender a todas as restrições simultaneamente. Na BTP, essa situação aparece na infactibilidade de metas quando o problema é particularmente difícil ou quando a atualização da restrição de função objetivo reduz significativamente o conjunto de soluções inteiras. Os algoritmos apresentados na Subseção 2.3.13 para resolução de infactibilidade de metas não apresentam uma sofisticação muito grande, pois apenas invertem metas e quando a inversão sozinha é insuficiente apelam para a liberação de metas, tornando a resolução da infactibilidade de metas um tanto quanto grosseira quando muitas metas são liberadas.

Um resolvedor de infactibilidade de metas perfeito precisaria testar todas as designações possíveis das variáveis com metas infactíveis e somente no insucesso de encontrar uma designação factível iniciar o processo de liberação de metas. Do ponto de vista prático, isso não pode ser realizado já que o problema de resolver o conflito de metas também é NP-Completo (encontrar uma solução factível para um problema de programação inteira). Ainda assim, um resolvedor mais eficiente do que o padrão (Subseção 2.3.13) foi implementado com auxílio do *branch-and-cut* do Cplex. A Figura 3.2 traz o algoritmo, a Tabela 3.1 lista os seus parâmetros e a Tabela 3.2 contém as suas variáveis. O resolvedor de infactibilidade de metas padrão não é removido e serve para sanar infactibilidades de metas que podem ser resolvidas com algumas poucas inversões de metas. Quando a busca permanece muitas iterações consecutivas na infactibilidade de metas sem que o número médio de variáveis com infactibilidade de metas seja reduzido, o resolvedor baseado em *branch-and-cut* é ativado. No *Passo* 0, um problema PIM residual é construído a partir da fase I do PL' aplicando algumas modificações. As variáveis binárias sem metas são tratadas como contínuas no intervalo de zero a um, pois, enquanto elas não têm metas, podem assumir qualquer valor. As variáveis com metas atendidas são fixadas no seu valor atual e as variáveis com metas não atendidas, conjunto G, são tratadas como variáveis

binárias. A função objetivo do PIM residual é a mesma da fase I do PL' e portanto o *branch-and-cut* minimiza a soma das distâncias às metas pré-estabelecidas. Note que o problema residual é restrito, pois todas as variáveis com metas atendidas foram inicialmente fixadas. O *branch-and-cut* é aplicado ao problema PIM residual por **TempoRConf** segundos no *Passo* 1. Nem sempre esse *branch-and-cut* tem sucesso, pois o tempo máximo pode expirar ou o problema residual pode não ter nenhuma solução factível. Nessa situação, tratada no *Passo* 4, algumas variáveis correntemente com metas atendidas são tratadas como potencialmente infactíveis e incluídas no conjunto G. No PIM residual elas deixam de ser fixas e passam a ser tratadas como variáveis binárias. O processo é repetido iterativamente para o conjunto G aumentado, mas o algoritmo limita o número máximo de variáveis com metas atendidas e que poderão ser tratadas como potencialmente infactíveis. Caso esse limite, **TotPerm,** seja alcançado o resolvedor baseado em *branch-and-cut* falha, *Passo* 3, e a BTP continua na infactibilidade de metas. Quando o resolvedor tem sucesso, *Passo* 2, a solução inteira encontrada por ele é convertida em metas para a próxima iteração da BTP e as listas tabu são atualizadas. As variáveis binárias do problema residual com valor zero recebem metas *DN* e as variáveis com valor um, metas *UP*. Os parâmetros **FracPermitida, FracPorIter, TempoRConf** e **multRR** precisam ser determinados empiricamente.

Tab. 3.1: Parâmetros do algoritmo de resolução de infactibilidade de metas por *branch-and-cut*.

Parâmetro	Descrição
FracPermitida	Fração de **TotVarsMetaFact**.
FracPorIter	Fração de **TotPerm.** São as variáveis com metas atendidas que são incluídas em G a cada iteração do algoritmo.
TempoRConf	Tempo máximo em segundos que o *branch-and-cut* é executado por iteração do algoritmo.
multRR	Multiplicador aplicado a **MediaInfact** para decidir quando aplicar o algoritmo.

3.2 Estratégias de longo prazo

Todas as abordagens de longo prazo aqui aplicadas à busca tabu paramétrica dependem da presença de um conjunto de referência. Uma vez bem definida a composição desse conjunto de referência, são desenvolvidas as idéias de diversificação e intensificação. Em seguida, duas técnicas de geração de centros que são derivadas de conceitos de *scatter search* (Glover et al., 2000) são apresentadas, e finalmente uma técnica de identificação de variáveis fortemente determinadas e variáveis consistentes conclui o capítulo.

Passo 0 Criação do PIM residual
se (**Iter** – **IterInicInfact**) $<$ **MediaInfact** * **multRR**
Vá para o *Passo* 5
senão

1. Faça uma cópia da fase I do PL' para criar o PIM residual.
2. Deixe como contínuas as variáveis binárias sem metas.
3. Fixe as variáveis binárias com metas atendidas em 0 ou 1 conforme o caso.
4. Deixe livre as variáveis binárias com metas não atendidas, tratando-as como binárias.
5. **TotPerm = FracPermitida*TotVarsMetaFact**
6. cont := 0

Passo 1 Aplicação do *branch-and-cut*
Resolva por meio de *branch-and-cut* o PIM residual por até **TempoRConf** segundos.

Passo 2 Sucesso, criação de novas metas a partir da solução do PIM residual
se uma solução foi encontrada no *Passo* 1

1. Variáveis em 0 recebem metas *DN* e as variáveis em 1 recebem metas *UP* no PL'.
2. Vá para o *Passo* 5

Passo 3 Falha do algoritmo
se (FracPorIter * cont $\geq$ 1**)**
Vá para o *Passo* 5

Passo 4 Expansão do conjunto G com variáveis potencialmente infactíveis.

1. Tome as **FracPorIter * TotPerm** variáveis com metas atendidas com maiores valores de resistência potencial a metas que ainda não foram incluídas em G. Deixe-as no PIM residual como variáveis binárias.
2. **cont** := **cont** +1
3. Vá para o *Passo* 1

Passo 5 Fim

Fig. 3.2: Algoritmo para resolver a infactibilidade de metas com uso de *branch-and-cut*.

Tab. 3.2: Variáveis do algoritmo de resolução de infactibilidade de metas por *branch-and-cut*.

Variável	Descrição
Iter	Iteração corrente da BTP.
IterInicInfact	Iteração em que a BTP transitou para a infactibilidade de metas.
TotVarsMetaFact	Número de variáveis com metas atendidas em **Iter**.
TotPerm	Máximo de variáveis com metas atendidas que podem ser incluídas em G.
MediaInfact	Média aritmética do número de variáveis com infactibilidade de metas nas (**Iter** – **IterInicInfact**) iterações.
cont	Controla o número de iterações do algoritmo.

3.2.1 Conjunto de referência

As abordagens de longo prazo desenvolvidas neste trabalho requerem a presença de pontos de referência dispersos pelas regiões visitadas pela busca no passado. Essas regiões são implicitamente definidas por subconjuntos de soluções. A criação desses subconjuntos pode ser conduzida a partir de um *conjunto de referência* $R = \left\{x^1, \ldots, x^b\right\}$ que não aceita réplicas e cujos elementos x^i, $i \in \{1, \ldots, b\}$ são soluções de boa qualidade que apresentam uma certa diversidade entre si. O uso do conjunto de referência é proposto por Glover (2006) e o desenvolvimento que segue nesta subseção é proposto pelo autor deste trabalho. Idealmente, o conjunto R deveria ser composto apenas de soluções inteiras-factíveis x'' encontradas pela BTP, mas como em muitas instâncias a BTP encontra um número reduzido de soluções inteiras, o conjunto R aceita soluções com infactibilidade inteira, aplicando o arredondamento para o inteiro mais próximo para cada componente x''_j de uma solução x'', de acordo com (3.4) para produzir x^i_j. Soluções com infactibilidade de metas são rejeitadas. A medida de infactibilidade inteira adotada foi a soma das distâncias ao inteiro mais próximo para todas as variáveis binárias, de acordo com a expressão (3.5). Essa medida de infactibilidade inteira foi utilizada para criar uma função objetivo modificada $\bar{x}_0$, expressão (3.6), que além de considerar o valor de função objetivo da fase II do PL', x''_0, incorpora uma componente relacionada à infactibilidade inteira. Essa expressão usa o valor da função objetivo do PL inicial com N' vazio, x_0^{PL} e a sua medida de infactibilidade inteira $infact\left(x_0^{PL}\right)$ como referências para fazer a soma ponderada do desvio de x''_0 em relação a x_0^{PL} e do desvio da infactibilidade inteira de x'', $infact\left(x''\right)$, em relação a $infact\left(x^{PL}\right)$. Quanto mais próximo de um estiver o valor do parâmetro ξ maior será a ênfase colocada no desvio da função objetivo e quanto mais próximo de zero, maior será a ênfase no desvio da infactibilidade inteira. A cardinalidade b do conjunto R é determinada empiricamente, bem como o parâmetro ξ de (3.6).

$$x_j^i = \begin{cases} 0 & \text{se} \quad x_j'' < 0,5 \\ 1 & \text{se} \quad x_j'' \geq 0,5 \end{cases} \tag{3.4}$$

$$infact\left(x''\right) = \sum_{j=1}^{n} \min\left(x_j'', 1 - x_j''\right) \tag{3.5}$$

$$\bar{x}_0 = \xi\left(\frac{x_0'' - x_0^{PL}}{x_0^{PL}}\right) + (1-\xi)\left(\frac{infact\left(x''\right) - infact\left(x^{PL}\right)}{infact\left(x^{PL}\right)}\right), \quad 0 \leq \xi \leq 1 \tag{3.6}$$

Há dois objetivos conflitantes que o conjunto R precisa atender. O primeiro deles diz respeito à qualidade dos elementos x^i, pois deseja-se que seus elementos tenham os menores valores possíveis de $\bar{x}_0^i$. O segundo objetivo está relacionado com uma medida de espalhamento espacial, uma vez que não é interessante que dois elementos em R sejam muito parecidos entre si. Para poder manter o espalhamento, a norma L_1 foi utilizada como medida de distância entre duas soluções, de acordo com a expressão (3.7). Também define-se a distância mínima em R como Ω na expressão (3.8).

$$\left\|x^i, x^k\right\| = \sum_{j=1}^{n} \left|x_j^i - x_j^k\right| \tag{3.7}$$

$$\Omega = \min\left\|x^i, x^k\right\|, \quad i, k \in \{1, .., b\}, \ i \neq k \tag{3.8}$$

Assume-se que os elementos em R estão classificados em ordem não-decrescente de valor $\bar{x}_0^i$, ou seja, $\bar{x}_0^1 \leq \bar{x}_0^2 \leq \ldots \leq \bar{x}_0^b$. Seja $Q = \{i_1, \ldots, i_r\} \subset \{2, \ldots, b\}$ um conjunto tal que se $i_c \in Q$ então existe um elemento $x^{i_d} \in R$ com $i_d \neq i_c$ e $\|x^{i_c}, x^{i_d}\| = \Omega$. Ou seja, Q representa os índices dos elementos em R cujas distâncias ao elemento mais próximo é a mínima em R, ao se excluir de Q a melhor solução x^1. Embora a distância utilizada seja simétrica, $\left\|x^i, x^k\right\| = \left\|x^k, x^i\right\|$, a cardinalidade de Q pode assumir o valor 1, pois por construção, x^1 não pode fazer parte de Q.

Quando x'' é uma solução candidata [1] a entrar no conjunto R, considera-se para efeito de análise o conjunto expandido $R^+ = R \cup \{x''\}$ e Ω^+ representa a distância mínima nesse conjunto. O conjunto R^+ também está ordenado tal que $\bar{x}_0^1 \leq \bar{x}_0^2 \leq \ldots \leq \bar{x}_0^p \leq \ldots \leq \bar{x}_0^{b+1}$ e x'' ocupa a posição p. Também há o conjunto $Q^+ = \{i_1, \ldots, i_s\} \subset \{2, \ldots, b+1\}$ tal que se $i_c \in Q^+$ então existe um elemento $x^{i_d} \in R^+$ com $i_d \neq i_c$ e $\|x^{i_c}, x^{i_d}\| = \Omega^+$.

No início da busca, todas as soluções x'' são inseridas no conjunto R, até a condição $|R| = b$ ser alcançada, aceitando muitas vezes soluções de baixa qualidade ou muito próximas entre si. Quando R atinge b elementos, a aceitação de x'' em R fica condicionada a um teste de qualidade e a um teste

[1] Aqui há um pequeno abuso de notação. Quando se fala da solução x'' fazendo parte do conjunto R, na verdade foi aplicado o arredondamento ao inteiro mais próximo com (3.4) para cada uma de suas componentes.

de espalhamento espacial. O teste de qualidade rejeita a solução x'' se simultaneamente $\bar{x}_0 > \omega\bar{x}_0^1$ e $\bar{x}_0 > \bar{x}_0^b$, observando-se que ω é um parâmetro tal que $\omega > 1$. Na Figura 3.3 há um exemplo de uma solução recusada. Havendo sucesso no teste de qualidade, o teste de espalhamento espacial é aplicado. Considera-se primeiramente o caso em que $\bar{x}_0 \geq \bar{x}_0^1$, com o conjunto R ainda não expandido em R^+. Se $\min_{i\in\{1,\ldots,b\}}\{\|x'', x^i\|\} < \Omega$, então a solução x'' é rejeitada, conforme o exemplo da Figura 3.4 pois não é conveniente que Ω diminua em R. Se $\min_{i\in\{1,\ldots,b\}}\{\|x'', x^i\|\} \geq \Omega$, considera-se a inclusão de x'' no conjunto R^+. Para que o conjunto R continue com cardinalidade b, alguma solução em Q^+ precisa ser removida. Os elementos em Q^+ estão empatados com a mesma distância mínima a algum elemento em R^+. Três critérios de desempate são usados para remover um elemento de Q^+.

1. O elemento x^{i_c} com $i_c \in Q^+$ é removido se $\max_{i\in\{1,\ldots,b+1\}}\{\|x^{i_c}, x^i\|\} < \min_{i_d\in Q^+\setminus\{i_c\}}\left\{\max_{i\in\{1,\ldots,b+1\}}\{\|x^{i_d}, x^i\|\}\right\}$. Caso haja empates, então esses elementos são incluídos em um conjunto $Q' \subset Q^+$e o teste 2 é aplicado aos elementos em Q' para desempatar.

2. O elemento x^{i_c} com $i_c \in Q'$ é removido se $\sum_{i\in\{1,\ldots,b+1\}}\|x^{i_c}, x^i\| < \min_{i_d\in Q'\setminus\{i_c\}}\left\{\sum_{i\in\{1,\ldots,b+1\}}\{\|x^{i_d}, x^i\|\}\right\}$. Caso haja empates, então esses elementos são incluídos em um conjunto $Q'' \subset Q'$ e o teste 3 é aplicado aos elementos de Q'' para desempatar.

3. O elemento x^{i_c} com $i_c \in Q''$ é removido se $\bar{x}_0^{i_c} > \max_{i_d\in Q''\setminus\{i_c\}}\left\{\bar{x}_0^{i_d}\right\}$. Se ainda persistir empate, o desempate é arbitrário.

Na análise anterior quando $\min_{i\in\{1,\ldots,b\}}\{\|x'', x^i\|\} = \Omega$, a solução candidata x'' torna-se x^p no conjunto temporário R^+. Nesse caso, $|Q^+| > |Q|$, pois o índice p de $\bar{x}^p$ em R^+ fará parte de Q^+, mas a sua aceitação em R não é garantida. Note também que nesse caso, Ω, a distância mínima em R não será aumentada, conforme o exemplo da Figura 3.5. Por outro lado, quando $\min_{i\in\{1,\ldots,b\}}\{\|x'', x^i\|\} > \Omega$ tem-se que $|Q^+| = |Q|$.

Quando $\bar{x}_0 < \bar{x}_0^1$, x'' é automaticamente aceita, pois nessa situação ocorre atualização da melhor solução do conjunto de referência. Nesse caso, como a solução entrando, x'', se transformará em x^1, não fará parte do conjunto Q^+. Essa é a única situação em que o valor Ω pode diminuir em R.

3.2.2 Intensificação e diversificação por análise de freqüência

O conjunto de referência R da Subseção 3.2.1 pode ser usado para construir uma matriz de freqüência proposta por Glover (2006) que correlaciona uma variável x_j com qualquer outra x_q, com x_j e x_q binárias e distintas. O objetivo é conhecer as freqüências com que as quatro possibilidades de designações $(x_j = 0, x_q = 0)$, $(x_j = 0, x_q = 1)$, $(x_j = 1, x_q = 0)$ e $(x_j = 1, x_q = 1)$ aparecem

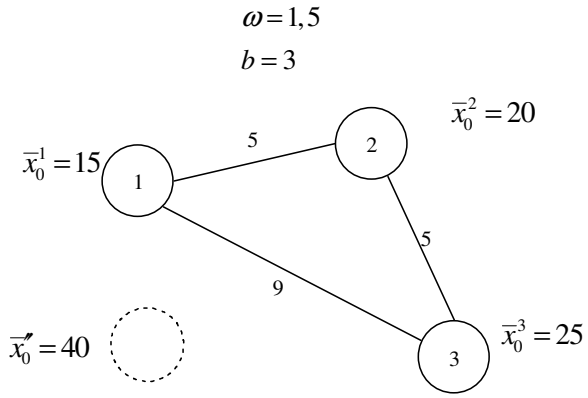

Fig. 3.3: Solução x'' é recusada por ter baixa qualidade.

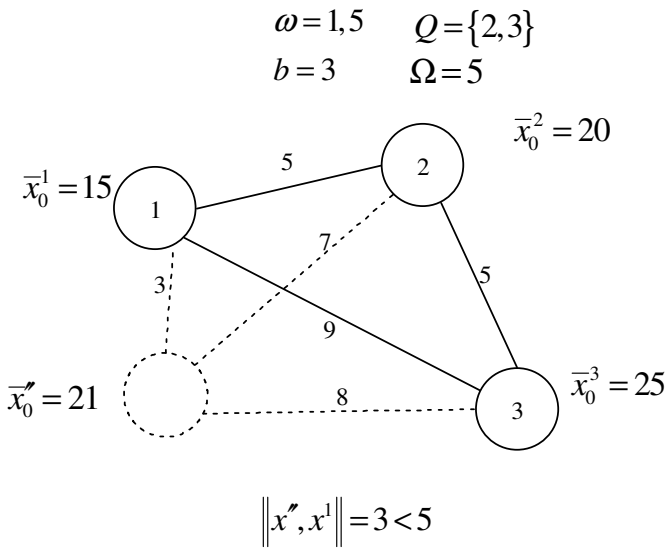

Fig. 3.4: A solução x'' é rejeitada porque a distância mínima, Ω, em R é reduzida de 5 para 3.

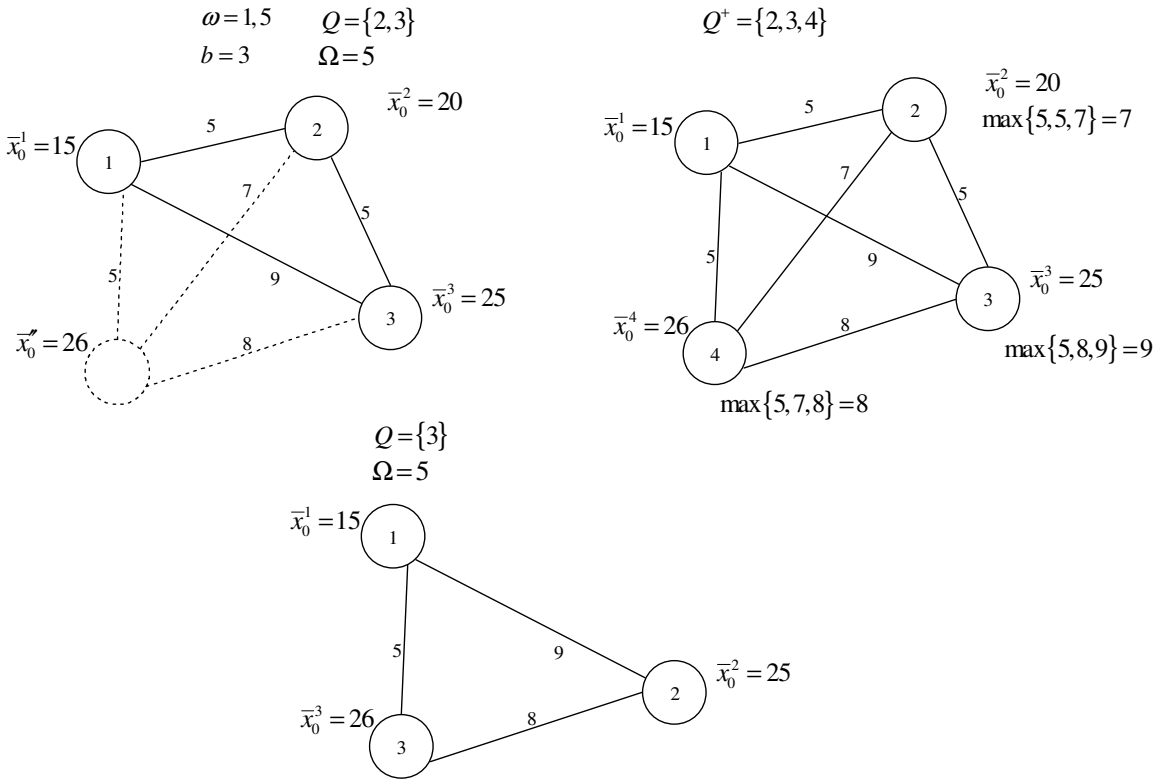

Fig. 3.5: A solução x'' é aceita em R.

quando medidas em relação às b soluções do conjunto R. Assim, a matriz de freqüência contém duas linhas e duas colunas para cada variável binária x_j de acordo com (3.9). O termo $x_{jq}(\gamma, \delta)$ contabiliza o número de soluções em R nas quais $x_j = \gamma$ e $x_q = \delta$, com γ e δ assumindo os valores 0 e 1. Estes valores representam as freqüências com que $x_q = 0$ e $x_q = 1$ ocorrem nas b soluções de R para os dois casos em que $x_j = 0$ e $x_j = 1$. Basta contar o número de ocorrências, já que b é o mesmo para todas as variáveis. A Tabela 3.3 representa um conjunto R com $b = 7$ elementos e $n = 6$ variáveis binárias, enquanto a Tabela 3.4 é a sua matriz de freqüência associada. Por exemplo, para as variáveis x_1 e x_2, pode-se computar a partir da Tabela Tabela 3.3 que as designações $(x_1 = 0, x_2 = 0)$, $(x_1 = 0, x_2 = 1)$, $(x_1 = 1, x_2 = 0)$ e $(x_1 = 1, x_2 = 1)$, ocorrem respectivamente 1, 2, 1 e 3 vezes em R. Esses valores aparecem na entrada [1,2] da Tabela 3.4. Algumas notas de implementação da matriz são apresentadas no Apêndice B.5.

$$\begin{aligned} x_j(0) &= (x_{jq}(0,0), x_{jq}(0,1)),\ q \in \{1,\ldots,n\} \\ x_j(1) &= (x_{jq}(1,0), x_{jq}(1,1)),\ q \in \{1,\ldots,n\} \end{aligned} \tag{3.9}$$

Tab. 3.3: Conjunto R com $b = 7$ elementos.

		Variáveis 0–1 (x_j)					
		1	**2**	**3**	**4**	**5**	**6**
Elementos em R	**1**	0	1	1	1	0	1
	2	1	1	0	1	1	0
	3	1	0	1	1	0	0
	4	0	1	0	1	1	0
	5	1	1	1	1	1	0
	6	0	0	1	0	0	1
	7	1	1	1	0	1	1

Tab. 3.4: Matriz de freqüência para o exemplo da Tabela 3.3.

	1		**2**		**3**		**4**		**5**		**6**	
1	3	0	1	2	1	2	1	2	2	1	1	2
	0	4	1	3	1	3	1	3	1	3	3	1
2	1	1	2	0	0	2	1	1	2	0	1	1
	2	3	0	5	2	3	1	4	1	4	3	2
3	1	1	0	2	2	0	0	2	0	2	2	0
	2	3	2	3	0	5	2	3	3	2	2	3
4	1	1	1	1	0	2	2	0	1	1	0	2
	2	3	1	4	2	3	0	5	2	3	4	1
5	2	1	2	1	0	3	1	2	3	0	1	2
	1	3	0	4	2	2	1	3	0	4	3	1
6	1	3	1	3	2	2	0	4	1	3	4	0
	2	1	1	2	0	3	2	1	2	1	0	3

Uma fase de intensificação ou diversificação pode ser conduzida usando a matriz de freqüência como base de dados. Seja S o conjunto dos índices das variáveis que estão sujeitas a uma condição de meta e x'_j o valor da meta para cada variável em S. Uma *medida de contagem* é definida pela expressão (3.10), na qual H pode representar o conjunto D das variáveis irrestritamente fracionárias na infactibilidade inteira e o conjunto G das variáveis com metas não atendidas na infactibilidade de metas. A medida $InScore_h\,(0)$ é a soma das freqüências para as quais $x_h = 0$ e $x_j = x'_j$. A medida $InScore_h\,(1)$ é a soma das freqüências para as quais $x_h = 1$ e $x_j = x'_j$. Assim pode-se saber o número de vezes que $x_j = x'_j$, $j \in S - \{h\}$ ocorre para $x_h = 0$ e para $x_h = 1$.

$$InScore_h\,(\gamma) = \begin{cases} \sum\limits_{j \in S-\{h\}} x_{hj}\left(0, x'_j\right), & h \in H \quad \text{se} \;\; \gamma = 0 \\ \sum\limits_{j \in S-\{h\}} x_{hj}\left(1, x'_j\right), & h \in H \quad \text{se} \;\; \gamma = 1 \end{cases} \tag{3.10}$$

Em uma fase de intensificação, na infactibilidade inteira, adota-se uma outra maneira de calcular as penalidades inteiras $IP_j\,(UP)$ e $IP_j\,(DN)$ com o uso de (3.11). Assim, quanto maior for a medida de contagem, menor será a penalidade inteira. Similarmente, na infactibilidade de metas, adota-se outras medidas para $GR_j\,(UP)$ e $GR_j\,(DN)$ de acordo com (3.12). Em uma fase de diversificação as medidas $IP_j\,(UP)$, $IP_j\,(DN)$, $GR_j\,(UP)$ e $GR_j\,(DN)$ podem ser calculadas com as expressões (3.13) e (3.14). Na diversificação quanto maior for a medida $InScore_h$, maior será a penalidade inteira.

A BTP passa a ter então três fases distintas que se alternam, a de curto prazo na qual os cálculos de penalidade inteira e resistência a metas são os estabelecidos no Capítulo 2, a de intensificação com os cálculos das expressões (3.11) e (3.12) e a de diversificação com os cálculos das expressões (3.13) e (3.14). A transição entre fases ocorre após um certo número consecutivo de iterações sem que haja atualização no conjunto de referência e é controlada pelos parâmetros α_cc, α_int e α_div. A busca inicia no curto prazo e ali permanece por $\alpha\_cc \times n$ iterações após o conjunto de referência ter estabilizado. Depois, sempre considerando iterações consecutivas com o conjunto de referência estável, transita para a intensificação permanecendo ali por $\alpha\_int \times n$ iterações, retorna para o curto prazo por mais $\alpha\_cc \times n$ e depois passa para diversificação por $\alpha\_div \times n$ iterações, fechando assim um ciclo completo da busca que se repete então iterativamente conforme a Figura 3.6.

$$IP_h\,(UP) = -InScore_h\,(1) \quad \text{e} \quad IP_h\,(DN) = -InScore_h\,(0) \tag{3.11}$$

$$GR_h\,(UP) = -InScore_h\,(1) \quad \text{e} \quad GR_h\,(DN) = -InScore_h\,(0) \tag{3.12}$$

$$IP_h\,(UP) = InScore_h\,(1) \quad \text{e} \quad IP_h\,(DN) = InScore_h\,(0) \tag{3.13}$$

$$GR_h\,(UP) = InScore_h\,(1) \quad \text{e} \quad GR_h\,(DN) = InScore_h\,(0) \tag{3.14}$$

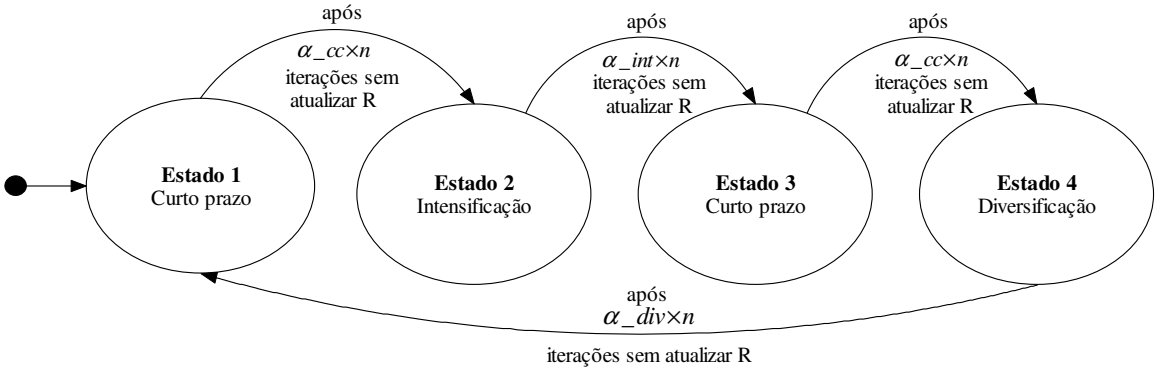

Fig. 3.6: Ciclo completo da busca tabu paramétrica com fases de curto e longo prazos.

3.2.3 Técnicas derivadas de *scatter search*

O uso das técnicas derivadas de *scatter search* propostas por Glover (2006) permite criar novos vetores de meta x' que são usados como pontos de partida para fases de intensificação e diversificação. Para esse fim deve-se evitar a geração de vetores x' que levem a uma fase da busca com reprodução de soluções já encontradas no passado. Para evitar esse cenário, dois conjuntos auxiliares que armazenam um histórico são usados. Como a composição do conjunto de referência R varia com a progressão da busca, um conjunto R^* que mantém o registro de todas as soluções que fizeram parte de R é estabelecido. Além de R^*, também é mantido o conjunto R' composto de todos os vetores meta x' associados às soluções x'' que entraram em R. Assim, se um novo vetor x' criado para iniciar uma fase de diversificação ou intensificação já existir em $R^* \cup R'$, deverá ser descartado e um novo deve ser gerado. A seguir são apresentadas as técnicas de geração de novos vetores x' com o seu PL' associado.

Intensificação e diversificação pela geração de centros x'

No método *scatter search* são geradas combinações lineares de pontos que definem uma sub-região particular de R. Na BTP, o foco se desloca para a geração de centros dessas sub-regiões em conformidade com a proposta da *scatter search* que evita visitar pontos em excesso. Para essa finalidade, uma sub-região coberta por R é implicitamente identificada ao selecionar um subconjunto $R(k)$ de k elementos escolhidos de R e que após serem reindexados são denotados por $R(k) = \{x^1, \ldots, x^k\}$. Os elementos de $R(k)$ são um a um escolhidos de R e uma vez que $R(i)$ tenha sido gerado para $1 \leq i < k$, a escolha de x^{i+1} que produz $R(i+1)$ é baseada na escolha do elemento em $R - R(i)$ que esteja mais próximo ou mais distante ao centro $C(i)$ definido de acordo com a expressão (3.15). Em uma fase de intensificação, a escolha de x^{i+1} é feita com a expressão (3.16) e na diversificação, é usada a expressão (3.17).

$$C(i) = \frac{1}{i} \sum_{h=1}^{i} x^h \tag{3.15}$$

$$x^{i+1} = \underset{x \in R - R(i)}{\arg\min} \left(\|x, C(i)\| \right) \tag{3.16}$$

$$x^{i+1} = \underset{x \in R - R(i)}{\arg\max} \left(\|x, C(i)\| \right) \tag{3.17}$$

Como o objetivo é utilizar a geração de centros várias vezes ao longo da busca, para que diferentes centros sejam gerados, foi feito uso de memória tabu para proibir que alguns elementos de R voltem a aparecer na geração de um novo $R(k)$. Assim são definidos dois conjuntos auxiliares *Tabu*1 e *Tabu*2.

O conjunto *Tabu*1 é a união de todos os $R(k)$ previamente gerados e o primeiro elemento a entrar em $R(k)$ vem de $R - Tabu1$ impedindo assim que o $R(k)$ sendo criado reproduza algum outro gerado no passado. No início da busca, então, $Tabu1 = \emptyset$ e cada vez que um novo $R(k)$ é gerado, todos os elementos são incluídos em *Tabu*1. A composição do conjunto *Tabu*2 depende do número de vezes que uma solução $x^i \in R$ apareceu em algum $R(k)$ e esse valor é denotado por $n(x^i)$. Quando $n(x^i)$ alcança o valor 2, uma solução de R passa a fazer parte de *Tabu*2, mas essa condição é desconsiderada para a melhor solução em R, x^1, que por aspiração não está sujeita à restrição *Tabu*2. Quando alguma solução em R é substituída, o valor $n(x^i)$ retorna ao valor inicial zero. A geração de $R(k)$ pode ser escrita na forma do algoritmo da Figura 3.7, lembrando que tanto *Tabu*1 quanto *Tabu*2 iniciam vazios antes do primeiro centro ser gerado.

Passo 1	Escolha x^1 como o elemento em $R - Tabu1$ com menor valor $\bar{x}_0^i$.
Passo 2	Para $i = 1, \ldots, k-1$, x^{i+1} é escolhido por $x^{i+1} = \underset{x \in R - R(i) - Tabu2}{\arg\min} (\|x, C(i)\|)$.
Passo 3	Faça $Tabu1 := Tabu1 \cup R(k)$. Para $i = 1, \ldots, k$, faça $n(x^i) := n(x^i) + 1$ e se $n(x^i) \geq 2$ adicione x^i a $Tabu2$.

Fig. 3.7: Algoritmo *RK*_1 para geração de $R(k)$.

Uma vez criado o conjunto $R(k)$ e identificado o seu centro $C(k)$, o vetor de metas x' que define a forma do PL' é criado ao arredondar as componentes de $C(k)$ ao inteiro mais próximo. O uso valores ímpares para k evita empates no processo de arredondamento por excluir o valor $0{,}5$. A criação de metas a partir de x' segue a expressão (3.18), todas as variáveis recebem metas *UP* ou *DN* e a partir daí uma nova fase de diversificação ou intensificação é iniciada. É importante ressaltar que caso o vetor de metas x' derivado do centro $C(k)$ já exista em $R^* \cup R'$, um outro deve ser gerado para que a busca não retorne a regiões já exploradas. A ênfase maior na diversificação pode ser conduzida ao mudar no *Passo* 2 o *argmin* para *argmax*.

$$\begin{array}{llll} \text{se} & x'_j = 0 & \text{inclua } j \text{ em} & N^- \\ \text{se} & x'_j = 1 & \text{inclua } j \text{ em} & N^+ \end{array} \tag{3.18}$$

Uma outra forma de gerar centros com maior variabilidade pode ser estabelecida com o auxílio da matriz $Match(p, q)$ em que $Match(p, q) = 1$ se as soluções x^p e x^q pertencentes a R apareceram simultaneamente em algum conjunto $R(k)$, e caso contrário, $Match(p, q) = 0$. Por convenção, assume-se que $Match(p, p) = 1$. A matriz $Match(p, q)$ é usada para admissão de um elemento x^p em $R(k)$ de acordo com $TabuMatch$ em (3.19). Ou seja, $TabuMatch$ é o conjunto de elementos de R que apareceram com cada um dos outros elementos de R em pelo menos um conjunto $R(k)$ gerado no passado. Assim, na geração de um novo conjunto $R(k)$, são excluídos os elementos que

pertencem a $TabuMatch$.

$$TabuMatch = \{x^p \in R \,|\, Match\,(p,q) = 1 \text{ para todo } x^q \in R\} \tag{3.19}$$

Além disso, relativo a um conjunto $R\,(i)$ que surge no processo de construção de $R\,(k)$ e a uma solução $x^p \in R - R\,(i)$ candidata a formar $R\,(i+1)$, define-se o valor $n\,(i|p)$ da expressão (3.20). Ou seja, $n\,(i|p)$ é o número de soluções $x^q \in R\,(i)$ tal que ambas x^p e x^q apareceram simultaneamente em algum $R\,(k)$. O elemento x^p não pode ser inserido em $R\,(i)$ caso faça parte do conjunto $Tabu\,(i)$ definido em (3.21). O parâmetro f é uma fração tal que $0 < f \leq 1$ e precisa ser calibrado.

$$n\,(i|p) = \sum_{x^q \in R(i)} Match\,(p,q) \tag{3.20}$$

$$Tabu\,(i) = \{x^p \in R - R\,(i) \,|\, n\,(i|p) > f \times |R\,(i)|\,\} \tag{3.21}$$

O critério de pertinência ao conjunto ***Tabu***2 que é $n\,(x^i) \geq 2$ no algoritmo RK_1 da Figura 3.7, muda para $n\,(x^i) \geq v\,(i)$. O parâmetro $v(i)$ controla o número de vezes que a solução x^i pode ser incluída em algum $R(k)$. Como Glover (2006) sugere usar $v\,(p) \geq v\,(q)$ se x^p tem avaliação superior a x^q, então foi aqui proposta a expressão (3.22) para cálculo de $v\,(i)$. A melhor solução x^1 recebe o valor $v\,(1) = Vmax$, enquanto que o pior elemento x^b recebe o valor $v\,(b) = Vmin$ observando que os parâmetros $Vmax$ e $Vmin$ precisam ser calibrados e que $b = |R|$.

$$v\,(i) = Vmax + \left\lceil \frac{(Vmin - Vmax)}{b-1} \times (i-1) \right\rceil \tag{3.22}$$

Com essas modificações, uma outra regra para geração de centros pode ser definida na qual o conjunto $Tabu1$ é substituído por $Tabu2 \cup TabuMatch$ no ***Passo*** 1. No início da busca os conjuntos ***TabuMatch*** e ***Tabu***(i) começam vazios e ***Match***(p,q) = 0 para todos os pares de soluções x^p e x^q em R. A cada execução do algoritmo da Figura 3.8, ***TabuMatch***, ***Tabu***(i) e ***Match***(p,q) são atualizados. Quando um elemento x^p em R é substituído, os valores ***Match***(p,q) e ***Match***$(q,\, p)$ voltam a zero para todo $x^q \in R$, $q \neq p$.

Passo 1	Escolha x^1 como o elemento em $R - (Tabu2 \cup TabuMatch)$ com menor valor $\bar{x}_0^i$.
Passo 2	Para $i = 1, \ldots, k-1$, x^{i+1} é escolhido por $x^{i+1} = \underset{x \in R - R(i) - (Tabu2 \,\cup\, Tabu(i))}{\arg\min} (\|x, C(i)\|)$
Passo 3	Para $i = 1, \ldots, k$, faça $n\,(x^i) := n\,(x^i) + 1$ e se $n\,(x^i) \geq v\,(i)$ adicione x^i a $Tabu2$.

Fig. 3.8: Algoritmo RK_2 para geração de $R(k)$.

Diversificação por complementação

Uma ênfase maior pode ser colocada na diversificação ao complementar o vetor x' produzido pelo algoritmo RK_1 ou RK_2 de geração de centros. Assim, adota-se $x'_j \leftarrow 1 - x'_j$ para $j \in N$.

3.2.4 Exploração de variáveis consistentes e fortemente determinadas

Variáveis consistentes em PIM 0–1 são aquelas que recebem um valor particular em uma proporção significativa de soluções de um conjunto $Y = \{x^1, \ldots, x^f\}$ escolhido. Variáveis fortemente determinadas são aquelas que apresentam grande deterioração no valor da função objetivo ou infactibilidade se o seu valor for alterado para as soluções do conjunto Y. É importante que os elementos do conjunto Y apresentem uma qualidade razoável e nas discussões que seguem, $Y = R$ ou $Y = R(k)$. Seja $Y^j_\gamma = \{x^i \in Y | x^i_j = \gamma\}$ o subconjunto de Y cujas soluções x^i apresentam $x^i_j = \gamma$. A freqüência de residência de uma variável x_j ao assumir um valor $\gamma \in \{0, 1\}$ no conjunto Y pode ser definida por (3.23) e o valor preferencial da variável x_j pode ser encontrado com (3.24). Como é sempre escolhida uma cardinalidade ímpar para Y, não ocorrem empates, ou seja, $freqRes_0(j) \neq freqRes_1(j)$.

$$freqRes_\gamma(j) = \left|Y^j_\gamma\right| \tag{3.23}$$

$$pref(j) = \begin{cases} 0 & \text{se} \quad freqRes_1(j) \leq freqRes_0(j) \\ 1 & \text{se} \quad freqRes_1(j) > freqRes_0(j) \end{cases} \tag{3.24}$$

O algoritmo da Figura 3.9 aqui proposto, cujas variáveis estão elencadas na Tabela 3.5 e os parâmetros na Tabela 3.6, procura por variáveis consistentes e fortemente determinadas num conjunto Y e as fixa no PL' atual. No *Passo* 1, as variáveis que não estão fixadas (**nãoFixadas)** são ordenadas em ordem não-crescente do máximo de suas freqüências de residência. No *Passo* 2, uma porcentagem (**maxAnalisar)** do número total das variáveis binárias é escolhida para ser analisada quanto à degradação na função objetivo. Essa fração envolve as variáveis com maiores valores de freqüência de residência. Para cada variável x_j selecionada, um PL'$_j$, criado a partir da fase II do PL' atual, é otimizado com a variável x_j sendo fixada no oposto do seu valor preferencial, ou seja, $x_j = 1 - pref(j)$. Assim a degradação na função objetivo, $\Delta_j = x_0^{\text{PL'}_j} - x_0^{\text{PL'}}$, pode ser computada em relação ao PL' atual. Reotimizações muito longas são evitadas ao limitar a execução em no máximo γ_2 iterações simplex. Na presença de infactibilidade, adota-se $\Delta_j = \infty$. As variáveis analisadas no *Passo* 2 que apresentarem maior deterioração Δ_j são escolhidas para o tratamento do *Passo* 3, usando para esse fim uma fração **maxFix** $\leq$ **maxAnalisar** do número total das variáveis binárias n. Uma *fase I auxiliar*, do PL de duas fases é criada para avaliar de forma paramétrica se as variáveis escolhidas no *Passo* 3 podem, de fato, ser fixadas. Cada x_j com $pref(j) = 1$ recebe uma meta *UP* e cada x_j com

$pref(j) = 0$ recebe uma meta *DN*. Variáveis que no PL' já estavam fixadas, também são fixadas na fase I auxiliar. Após a otimização da fase I auxiliar, somente as variáveis que tiverem as metas atendidas podem ser fixadas no PL' atual e é claro que se nenhuma variável puder ser fixada, então o algoritmo falhou. No conjunto Ynão podem restar elementos que reflitam a situação oposta de uma variável que foi fixada, assim se uma variável foi fixada tal que $x_j = \gamma$, então os elementos do conjunto Y que apresentarem $x_j^i = 1 - \gamma$, são removidos. Os parâmetros a serem calibrados são **maxAnalisar** e **maxFix.**

Tab. 3.5: Variáveis do algoritmo da Figura 3.9.

nãoFixadas	Variáveis binárias que não foram fixadas no PL' por execuções anteriores do algoritmo.
totAnalisar	Número de variáveis que serão analisadas pelo algoritmo.
totFix	Número máximo de variáveis que serão fixadas pelo algoritmo.

Tab. 3.6: Parâmetros do algoritmo da Figura 3.9.

maxAnalisar	Porcentagem das n variáveis binárias que serão analisadas pelo algoritmo.
maxFix	Porcentagem máxima das n variáveis que serão fixadas pelo algoritmo.
γ_2	Número máximo de iterações simplex permitido na reotimização do PL'$_j$.

Quando um algoritmo progressivamente restringe as variáveis consistentes e fortemente determinadas aos seus valores preferenciais, novas variáveis dentre as remanescentes apresentarão avaliações que as qualificam mais decisivamente como fortemente determinadas e consistentes. Assim, o algoritmo **FixaVars** da Figura 3.9 não deve ser aplicado uma única vez, mas deve estar subordinado a um algoritmo **Coordenador** que controla o número máximo de variáveis que podem ser fixadas e como são compostos os conjuntos Y passados para **FixaVars**. Quando o número de variáveis fixadas for muito grande, o algoritmo deve retroceder a estágios anteriores em que menos variáveis estavam fixadas. Ao utilizar uma outra seqüência de conjuntos Y, embute-se um efeito diversificador na classificação das variáveis consistentes e fortemente determinadas, favorecendo a exploração do espaço de busca de modo eficiente. Imediatamente antes da aplicação de uma fase de intensificação definida nos moldes da Subseção 3.2.2, o algoritmo **Coordenador** assume o controle e o seu comportamento dinâmico está representado na Figura 3.10. No nível 0, quando nenhuma variável foi ainda fixada, a partir do conjunto de referência R_0, **maxRk** conjuntos $R(k)$ são criados, observando-se que para o exemplo foi estabelecido **maxRk**=3. Adota-se $Y = R_{0,1}(k)$ sobre o qual é aplicado o algoritmo **FixaVars** enquanto $R_{0,2}(k)$ e $R_{0,3}(k)$ são guardados para serem tratados no futuro. Com

Passo 1	Usando o conjunto Y, ordene as variáveis do conjunto **nãoFixadas** em ordem não-crescente de $\max\{freqRes_0(j), freqRes_1(j)\}$.
Passo 2	1. **totAnalisar** := min(**nãoFixadas, maxAnalisar** $\times\ n$). 2. Selecione as primeiras **totAnalisar** variáveis ordenadas no *Passo* 1. Para cada uma dessas variáveis x_j, reotimize o PL'$_j$ por até γ_2 iterações simplex fixando a variável x_j no valor $1 - pref(j)$. Calcule a degradação $\Delta_j = x_0^{\text{PL'}_j} - x_0^{\text{PL'}}$. Se o PL'$_j$ for infactível, $\Delta_j = \infty$.
Passo 3	1. Crie uma fase I do PL de duas fases (fase I auxiliar), estabelecendo metas para as **totFix** := min(**totAnalisar, maxFix** $\times\ n$) variáveis com maiores valores de degradação encontradas no *Passo* 2. Cada x_j com $pref(j) = 1$ recebe uma meta *UP* e cada x_j com $pref(j) = 0$ recebe uma meta *DN*. Variáveis que no PL' já estavam fixadas, também são fixadas nessa fase I auxiliar. 2. Otimize a fase I auxiliar.
Passo 4	Fixe no PL' atual todas as variáveis que tiveram suas metas atendidas na fase I auxiliar do *Passo* 3. Para cada variável fixada, $x_j = \gamma$, remova do conjunto Y cada elemento x^i tal que $x_j^i = 1 - \gamma$. Se nenhuma variável foi fixada, o algoritmo termina com insucesso, caso contrário termina com sucesso.

Fig. 3.9: FixaVars - Algoritmo de descoberta de variáveis consistentes e fortemente determinadas.

o progresso da busca, $R_{0,1}(k)$ receberá novos elementos evoluindo para $R_{1,1}$. Quando uma nova fase de intensificação for iniciar, a mesma seqüência de operações aplicada em R_0 é reaplicada em $R_{1,1}$, gerando os conjuntos $R_{1,1,1}(k)$,$R_{1,1,2}(k)$ e $R_{1,1,3}(k)$. Os conjuntos $R_{1,1,2}(k)$ e $R_{1,1,3}(k)$ são guardados para análise futura e adota-se $Y = R_{1,1,1}(k)$ sobre o qual é aplicado o algoritmo **FixaVars**. Há um limite **nivelMax** a partir do qual as ramificações pela geração de conjuntos $R(k)$ são interrompidas. No exemplo, **nivelMax=**1, então a partir do nível 2, não são mais gerados conjuntos e adota-se $Y = R_{2,1,1}$, $Y = R_{3,1,1}$ e assim sucessivamente, fixando mais e mais variáveis à medida que **FixaVars** é invocado antes do início de cada fase de intensificação. Eventualmente, num certo nível q o algoritmo **FixaVars** pode não conseguir fixar mais nenhuma variável ou a fração máxima de variáveis que podem ser fixadas no PL', **fracaoLimite**, será atingida. Nesse ponto, ocorre um regresso na árvore, as variáveis que devem ficar fixadas são apenas aquelas presentes no conjunto **fixadas**$_{1,1}$, o conjunto $R_{1,1,2}(k)$ é recuperado adotando-se $Y = R_{1,1,2}(k)$ e o algoritmo **FixaVars** é aplicado. Quando a árvore inteira tiver sido percorrida, o algoritmo é reaplicado até que em R_0 se esgotem as

possibilidades de geração de conjuntos $R\,(k)$. Como os conjuntos $R\,(k)$ são sempre distintos, quando $R_{0,2}\,(k)$ for recuperado após o tratamento completo da subárvore enraizada no nó $R_{1,1}$, as variáveis classificadas como fortemente determinadas e consistentes, **fixadas**$_{1,2}$, provavelmente serão ligeiramente diferentes daquelas em **fixadas**$_{1,1}$, levando a uma exploração efetiva do espaço de busca. Os detalhes da implementação do algoritmo **Coordenador** são apresentados na Subseção 3.2.5.

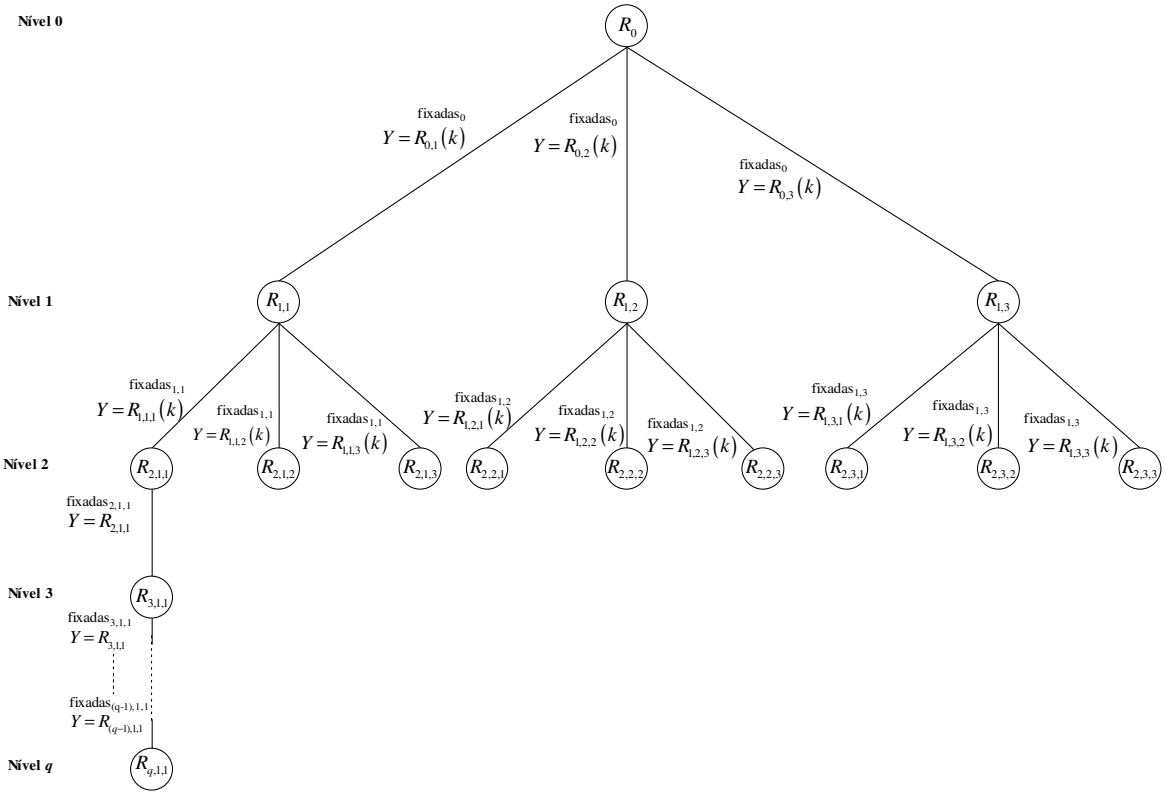

Fig. 3.10: Comportamento dinâmico do algoritmo **Coordenador**.

3.2.5 Notas de implementação do algoritmo Coordenador

Ao invés de apresentar uma sugestão de implementação naturalmente recursiva para o algoritmo **Coordenador** da Subseção 3.2.4, optou-se por uma implementação iterativa com uso de pilha explícita apresentada na Figura 3.12. A Tabela 3.8 contém as variáveis do algoritmo e a Tabela 3.7 contém os parâmetros já discutidos na análise da árvore da Figura 3.10. A pilha é composta de uma trinca de valores **{**Y**, fixadas, nivel}** representando respectivamente um conjunto R ou um subconjunto $R\,(k)$, o conjunto de variáveis atualmente fixadas e o nível de profundidade em que se encontra na árvore. O

Passo 1 avalia se o regresso na árvore deve ser realizado em caso de excesso de variáveis fixadas. O *Passo* 2 cuida da geração e empilhamento dos conjuntos $R(k)$ para níveis próximos à raiz da árvore e para os níveis mais profundos efetua apenas o empilhamento de R. O *Passo* 3 trata a situação da pilha vazia, e para tanto, retorna ao nível zero e libera todas as variáveis. O *Passo* 4 é o responsável pela aplicação de **FixaVars** para os dados disponíveis no topo da pilha.

Para o exemplo da Figura 3.10, no início, quando não há variáveis fixadas, o teste do *Passo* 1 não tem sucesso, no *Passo* 2 são gerados os conjuntos $R(k)$ com o empilhamento de $R_{0,3}(k)$, $R_{0,2}(k)$ e $R_{0,1}(k)$ nessa ordem. O conjunto $R_{0,1}(k)$, que fica no topo da pilha é removido no *Passo* 4 com a aplicação do algoritmo **FixaVars**. Quando a nova fase de intensificação for iniciar, $R_{0,1}(k)$ terá evoluído para um conjunto de referência $R_{1,1}$. A aplicação dos passos do algoritmo segue a mesma seqüência com o empilhamento de $R_{1,1,3}(k)$**,** $R_{1,1,2}(k)$ e $R_{1,1,1}(k)$. No *Passo* 4, $R_{1,1,1}(k)$ é desempilhado com a aplicação de **FixaVars**. A partir do nível 2, no *Passo* 2 não são mais gerados os conjuntos $R(k)$ e os conjuntos $R_{2,1,1}$, $R_{3,1,1}$, ..., $R_{q,1,1}$, nessa seqüência, são um a um inseridos na pilha e removidos dela a cada execução do algoritmo, culminando com a situação presente na pilha da Figura 3.11 que representa o momento anterior ao retrocesso na árvore que ocorre no nível q. O algoritmo **FixaVars** falha no *Passo* 4, para $Y = R_{q,1,1}$, volta ao *Passo* 3 e segue imediatamente ao *Passo* 4 desempilhando $R_{1,1,2}(k)$ sobre o qual é aplicado **FixaVars**. Se acontecer da pilha ficar vazia no *Passo* 3, quando uma nova fase de intensificação for iniciar, terá inicio a construção de uma nova árvore caso ainda seja possível extrair conjuntos $R(k)$ de R_0.

$R_{q,1,1}$	$\text{fixadas}_{(q-1),1,1}$	q	⇦ Topo da pilha
$R_{1,1,2}(k)$	$\text{fixadas}_{1,1}$	2	
$R_{1,1,3}(k)$	$\text{fixadas}_{1,1}$	2	
$R_{0,2}(k)$	fixadas_{0}	1	
$R_{0,3}(k)$	fixadas_{0}	1	

Fig. 3.11: Exemplo de pilha para o algoritmo Coordenador.

Tab. 3.7: Parâmetros do algoritmo Coordenador.

fracaoLimite	Fração limite das n variáveis binárias que podem ser fixadas pelo algoritmo.
nivelMax	Profundidade da árvore até a qual a geração de conjuntos $R(k)$ é conduzida.
maxRk	Número máximo de conjuntos $R(k)$ que podem ser gerados num nó da árvore.

Tab. 3.8: Variáveis do algoritmo Coordenador.

fracaoFixada	Fração das n variáveis binárias que foram fixadas pelo algoritmo.
nivelAtual	Profundidade em que o algoritmo se encontra na árvore.
topo	Topo da pilha.

Passo 1 **se fracaoFixada** $>$ **fracaoLimite**
Vá para o *Passo* 3.

Passo 2 **se** (**nivelAtual** $\leq$ **nivelMax**)

1. Gere até **maxRk** subconjuntos $R\,(k)$ a partir de R.
2. Empilhe os conjuntos na ordem inversa da que foram gerados e acompanhados do conjunto das variáveis fixadas até aqui e de **nivelAtual+1**, compondo as trincas **{**$R(k)$**, fixadas, nivelAtual+1}**.

senão

1. Empilhe o conjunto R atual acompanhado do conjunto das variáveis fixadas até aqui e de **nivelAtual+1**, compondo a trinca **{**R**, fixadas, nivelAtual+1}**.

Passo 3 **se** a pilha estiver vazia

1. Libere todas as variáveis fixadas.
2. **nivelAtual** := 0
3. Vá para o *Passo* 5.

Passo 4

1. Atualize o PL' fixando as variáveis de acordo com **topo.fixadas**.
2. R:= **topo**.Y
3. Tente fixar novas variáveis com o procedimento **FixaVars** da Figura 3.9.
4. **nivelAtual** := **topo.nivel**
5. Remova o topo da pilha.
6. **se** nenhuma variável nova foi fixada
   Vá para o *Passo* 3.

Passo 5 Fim

Fig. 3.12: Algoritmo Coordenador.

Capítulo 4

Resultados Computacionais em Instâncias da Miplib

Os testes foram realizados em um PC com processador Intel Pentium IV de 3,2 GHz, 3 Gbytes de memória e sistema operacional Fedora 4. O resolvedor de programação linear utilizado foi o do ILOG-Cplex 10.0 por meio da interface OsiCpx do COIN (COIN-OR 2009). Os algoritmos foram implementados em linguagem C++ com o compilador gcc 4.0.2. O critério de parada foi o tempo computacional de 600s para calibrações de curto prazo, 1200s e 2400s para calibrações de longo prazo não acuradas e finas, respectivamente, e 3600s para resultados finais. O tempo começou a ser medido após o pré-processamento e resolução do PL original. Os testes computacionais foram realizados sobre um conjunto de 78 instâncias coletadas da Miplib 2003 e Mittelmann (2003). Essas instâncias foram as mesmas usadas por Achterberg e Berthold (2007), excluídas as instâncias que envolvem variáveis inteiras gerais. A apresentação dos resultados é dividida em três partes. A primeira se concentra na escolha das melhores abordagens do curto prazo. Na seqüência, as extensões aplicáveis ao curto prazo são validadas e o capítulo é encerrado com a apresentação do desempenho das estratégias de longo prazo.

4.1 Resultados com uso de memória de curto prazo

O estabelecimento de uma implementação de referência da busca tabu paramétrica demandou a validação da eficiência de componentes que poderiam ser habilitados ou desabilitados. Outros elementos exigiram a escolha de uma abordagem mais eficiente, como o uso de DM, RB ou ACVO para o cômputo da resistência a metas e penalidade inteira. Também foi preciso decidir pelo uso de frações estáticas ou abordagem adaptativa para o controle da cardinalidade dos conjuntos G_p, G_s e D_0. Em seguida, a BTP foi comparada com as heurísticas do CPLEX 10 e com a *Objective Feasibility*

Pump (Achterberg e Berthold, 2007).

4.1.1 Calibração dos parâmetros de curto prazo

Vários parâmetros da busca precisaram ser calibrados, como a memória embutida no modelo por meio de M_j dinamicamente atualizado, a duração tabu, o parâmetro ε que controla a restrição de função objetivo, as frações que determinam as cardinalidades dos conjuntos G_p, G_s e D_0, os parâmetros do *Reliability Branching* e das expressões de escolha preferencial.

Os pesos M_j da expressão (2.7) dependem de dois parâmetros. O primeiro deles, NI, que controla o número de iterações em que uma meta *UP* ou *DN* tem maior influência, foi definido em 64 e foi suficiente para melhorar o desempenho da busca em relação ao uso de M_j constante. Valores maiores que esse não apresentaram melhores resultados. Para o segundo parâmetro, testes com $r \in \{0{,}10;\ 0{,}15;\ 0{,}20\}$ apresentaram resultados semelhantes, com ligeira vantagem para $r = 0{,}20$, que foi o valor escolhido. Valores inferiores a $0{,}10$ geram valores de M_j muito próximos entre si. Por outro lado, para $r > 0{,}20$, os valores de M_j assumem uma magnitude muito elevada, principalmente nas primeiras iterações seguintes à aplicação de uma meta, conforme a Figura 4.1.

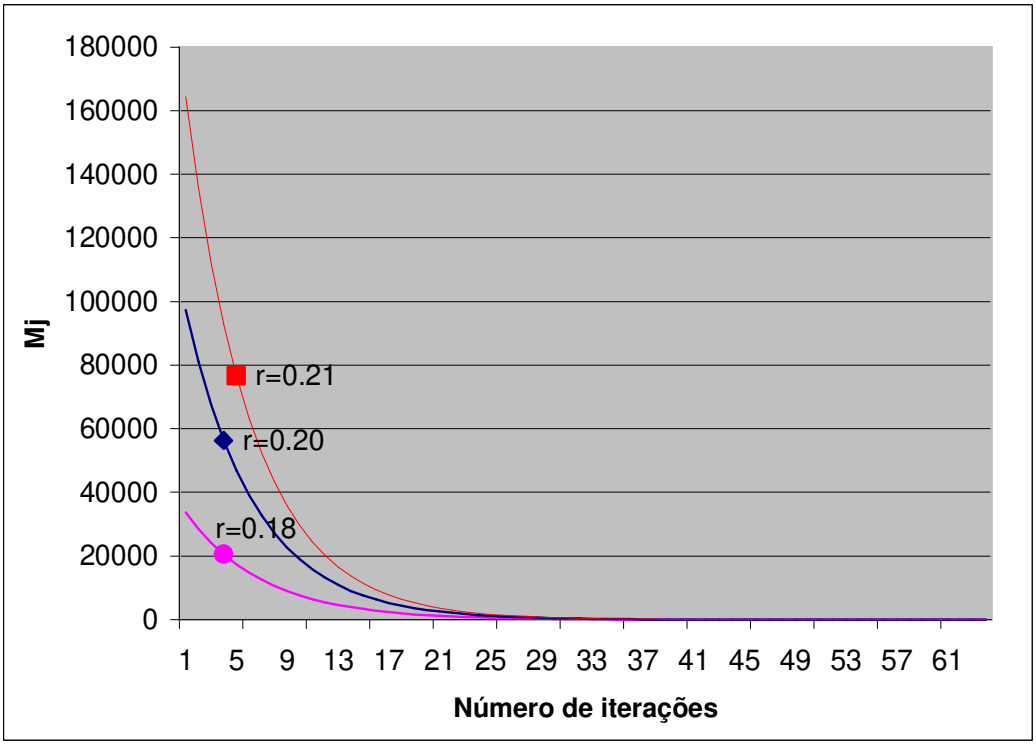

Fig. 4.1: Variação de r na expressão (2.7) e os valores de M_j ao longo das iterações.

O parâmetro ε da restrição de função objetivo $cx + dy \leq x_0^* - \varepsilon$ em que x_0^* é o valor da melhor solução conhecida deve ser pequeno o suficiente para não perder soluções inteiras-factíveis ainda melhores. Mas às vezes os coeficientes das variáveis na função objetivo têm magnitude muito grande quando comparados com um valor de ε pequeno. Nessa situação, dificuldades numéricas podem ocorrer impedindo que um ε pequeno tenha efeito no PL' atual e a atualização da restrição de função objetivo não torna infactível uma solução incumbente recém-encontrada. Via de regra foi usado $\varepsilon = 1$, exceto para as instâncias *modglob* e *glass4* para as quais foram usados respectivamente $\varepsilon = 1.000$ e $\varepsilon = 450.000$. Uma maneira de usar um único valor ε pode se estabelecida ao normalizar a restrição de função objetivo ao dividi-la por $|x_0^*|$, $\frac{x_0^* - (cx+dy)}{|x_0^*|} \geq \varepsilon$.

Na calibração dos parâmetros da duração tabu apresentados na Subseção 2.3.11, considerou-se $k_1 \in \{0{,}05;\ 0{,}10;\ 0{,}15;\ 0{,}20\}$ e $k_3 \in \{0{,}10;\ 0{,}15;\ 0{,}20;\ 0{,}25;\ 0{,}30\}$, formando pares tais que $k_1 < k_3$. Para valores $k_1 < 0{,}05$, a busca tende a ciclar e com $k_3 > 0{,}10$ é comum todas as variáveis com infactibilidade de metas tornarem-se tabu, prejudicando a busca. Os melhores valores encontrados foram $k_1 = 0{,}05$ e $k_3 = 0{,}10$. O parâmetro k_2 é menos sensível e é desejável que seja pequeno para evitar que *MinDuração* atinja *MaxDuração* prematuramente. O valor adotado foi $k_2 = 0{,}01$. Adotou-se também $MaxLimpaTabuIter = 2.000$ iterações.

Para determinar a cardinalidade dos conjuntos G_p, G_s e D_0, as frações mais adequadas f_d, f_p e f_s para as expressões (2.21), (2.22) e (2.23) da abordagem por frações estáticas foram testadas para os valores $\{0{,}05;\ 0{,}10;\ 0{,}15;\ 0{,}20;\ 0{,}25;\ 0{,}30\}$. O parâmetro f_d escolhido foi 0,05. Manter f_d grande quando há uma transição da infactibilidade inteira para a infactibilidade de metas é prejudicial para a busca. Ao estabelecer muitas metas adicionais quando o PL' já possui um número substancial de metas pode demandar muitas iterações para resolver uma infactibilidade de metas. A inversão das metas das variáveis do conjunto G_p muda a informação acumulada pela busca, mas a remoção das metas das variáveis do conjunto G_s descarta informações presentes no PL' sendo uma maneira pobre de resolver uma infactibilidade de metas. Os parâmetros f_p e f_s apresentaram melhor resultado para 0,05. Acima desse valor, a inversão de muitas metas bem como a liberação de muitas variáveis de suas metas dificulta a descoberta de soluções inteiras-factíveis. O número T^0 de variáveis potencialmente infactíveis foi mantido em 1, apenas para contornar a situação em que todas as variáveis diretamente infactíveis em relação a metas são classificadas como tabu. Elevar o valor de T^0 não apresentou melhoria mesmo quando assumiu até 25% do total de variáveis binárias, $T^0 = 0{,}25n$. Os valores mínimos de cardinalidades foram mantidos em 1, $d_{\min} = g_{p_{\min}} = g_{s_{\min}} = 1$.

A análise empírica dos resultados com frações estáticas naturalmente conduziu ao desenvolvimento do controle adaptativo da cardinalidade dos conjuntos G_p, G_s e D_0 a partir de algumas observações importantes. A fração f_d deve começar com um valor grande para acelerar o estabelecimento de metas no início da busca, diminuir quando a busca transita para a infactibilidade de metas e deve

ser aumentada quando a busca permanece muitas iterações na infactibilidade inteira. Os conjuntos G_p e G_s devem ser mantidos com cardinalidade baixa, a inversão de poucas metas por iteração, uma ou duas, por exemplo, é recomendado. A liberação de variáveis de suas metas também deve ser mantida em um patamar baixo, numa tentativa de sanar a infactibilidade de metas com a menor perturbação possível no PL'. Com essas observações, o critério adaptativo recebeu os parâmetros $g_{p_{\min}} = 1$, $g_{s_{\min}} = 0$, $\delta_2 = 1{,}05$, $\delta_1 = 0{,}5$, $f_{d_{\max}} = 0{,}3$, $k_4 = 0{,}25$, $g_{\max} = 3$ e $f_s = 0{,}05$.

Os parâmetros de *Reliability Branching* usados foram $\eta_{rel} = 8$, $\lambda = 8$ e $\gamma = 10$. O parâmetro γ serve principalmente para evitar longas reotimizações duais simplex. Valores de η_{rel} e λ maiores começam a degenerar o *Reliability Branching* em um *Full Strong Branching* (Achterberg et al., 2005), e como muito processamento é gasto em reotimizações simplex, menos iterações da busca tabu paramétrica são realizadas, encontrando então menos soluções para um dado limite de tempo. As combinações de uma expressão de escolha preferencial (2.10), (2.11) ou (2.12) com uma das regras de cálculo de penalidade como DM, RB ou ACVO estão resumidas na Tabela 4.1. Em (2.10) foi usado $w = 0{,}01$, apenas para evitar que o produto presente na expressão se anule quando $f_j^+ = f_j^-$, já em (2.11) $\mu = 5/6$ foi o melhor valor encontrado para $\mu \in \left\{0, \frac{1}{6}, \frac{2}{6}, \frac{3}{6}, \frac{4}{6}, \frac{5}{6}, 1\right\}$.

Tab. 4.1: Expressão de escolha preferencial para cada técnica de criação de metas.

Cálculo de penalidade inteira e resistência a metas	Expressão de escolha preferencial
DM	Expressão (2.10)
RB	Expressão (2.11)
ACVO	Expressão (2.12)

4.1.2 Validação das estratégias de curto prazo

As estratégias de curto prazo utilizadas na busca foram validadas com relação à eficiência antes de serem definitivamente mantidas para a análise dos resultados finais. A implementação de referência (**ImpRef**) usada para conduzir essas análises envolveu o uso da técnica ACVO para o cálculo da infactibilidade inteira e infactibilidade de metas. O critério de aspiração por resistência ficou ativado, o controle da cardinalidade dos conjuntos G_p, G_s e D_0 foi realizado pela técnica de controle adaptativo e o uso de M_j dinamicamente atualizado pela expressão (2.7) ficou ativado. Três testes foram conduzidos ao fazer alterações na implementação de referência. O primeiro (**M_cte**) desabilitou o M_j dinâmico, o segundo **(SemAspir)** desabilitou o critério de aspiração por resistência e o terceiro **(FracEstática)** trocou a estratégia adaptativa para controle das cardinalidades dos conjuntos G_p, G_s

e D_0 pelo uso de frações estáticas.

O teste estatístico de classificação com sinal de Wilcoxon (Mosteller e Rourke, 1973) foi aplicado para comparar as esperanças $E_X[VO]$ e $E_Y[VO]$ em que Xe Y denotam duas estratégias e VO é a variável aleatória que representa o valor da função objetivo de um conjunto de instâncias. Esse teste é muito útil para comparar o desempenho de duas heurísticas (Golden e Stewart, 1985) e consiste em calcular a diferença dos valores aleatórios de função objetivo (VO) para cada instância. Classificações são designadas para cada diferença de pares de valores e a hipótese nula, $E_X[VO] = E_Y[VO]$, é testada com as hipóteses alternativas $E_X[VO] > E_Y[VO]$ ou $E_X[VO] < E_Y[VO]$ com nível de confiança de 0,05. Quando não é encontrada uma solução factível para uma instância, ela entra no teste assumindo valor infinito de função objetivo. Quando ocorreu empate com o uso desse teste, o desempate se fez em prol da implementação que falhou[1] em menos instâncias. Com a persistência do empate, o critério de desempate foi em favor da abordagem que apresentou o maior número de instâncias com desvio não superior a 5% em relação à melhor solução conhecida. Alguns fatores dificultam a avaliação com uso do desvio médio em relação aos melhores valores de função objetivo conhecidos. A natureza das instâncias utilizadas é bastante diversificada assim como os melhores valores de função objetivo conhecidos. As instâncias de acc-0 até acc-6, cujos valores ótimos de função objetivo valem zero não podem ser incluídas numa análise desse tipo. Também há instâncias para as quais nenhuma solução factível é encontrada por uma abordagem e também precisam ser excluídas dessa análise. Ainda assim, os desvios médios são apresentados e podem ser úteis quando os critérios anteriores mostram-se inconclusivos.

Os resultados das quatro execuções estão reunidos na Tabela 4.2. As primeiras duas colunas são relativas aos nomes das instâncias e dos melhores valores de soluções conhecidos. Os nomes de instâncias apresentados em caracteres itálicos indicam que a solução ótima é desconhecida. Cada uma das quatro abordagens é apresentada em duas colunas. A primeira representa o valor de função objetivo encontrado e a segunda é o desvio calculado com $desv = 100 \times \frac{x_0 - x_0^*}{|x_0^*|}$ em que x_0^* é o melhor valor conhecido de função objetivo e x_0 é o valor encontrado pela heurística. A presença do símbolo '–' nas colunas **Objetivo** e **Desv** indica falha da heurística. Para as instâncias de acc-0 a acc-6, o símbolo '–' na coluna **Desv** indica a impossibilidade do cálculo do desvio. A linha **Média** no final da tabela traz o desvio médio para as instâncias em que uma solução foi encontrada, a linha **Média(65)** exibe o desvio médio considerando apenas as instâncias em que as 4 execuções conseguiram encontrar uma solução inteira-factível e a linha **Falhas** indica em quantas instâncias uma execução falhou. Os resultados em que uma execução alcançou a solução ótima estão destacados em negrito. O símbolo 0^+ indica que o desvio ficou abaixo de 0,5%. O tempo computacional foi de

[1] A falha de uma heurística nesta dissertação significa que a execução não encontrou nenhuma solução inteira-factível para uma dada instância durante um limite de tempo estabelecido.

3600s e essas convenções foram adotadas em todos os resultados apresentados nesta dissertação. A Tabela 4.3 apresenta os mesmos resultados agregados por intervalos de desvio. Para cada abordagem há duas colunas, a primeira representa o número de instâncias e a segunda é a porcentagem dentre as 78 consideradas. Quando uma execução encontra uma solução não-ótima para uma das instâncias de acc-0 a acc-6, são contabilizadas na linha 'Não ótimas em ACCs'.

O primeiro teste conduzido avaliou a eficácia do uso do M_j na fase I do PL'. O uso de M_j dinâmico estabelecido pela expressão (2.7) foi confrontado contra a situação limite em que $M_j = 1$ (M_cte) para todas as variáveis, bastando para isso adotar $r = 0{,}0$ em (2.7). O uso de M_j variável favorece a descoberta de soluções inteiras, em apenas 5 instâncias a ImpRef falhou enquanto que a busca falha em 12 instâncias quando $M_j = 1$. Além disso, o desvio médio é melhor com M_j dinâmico em ImpRef, 93% contra 401% para M_cte. É válido ressaltar que a execução com $M_j = 1$ encontrou solução para a instância momentum1 com desvio de 94%. Nas demais variações da BTP de curto prazo com uso da estratégia ACVO, não houve sucesso em encontrar soluções factíveis para a instância momentum1. No teste de Wilcoxon, ImpRef também foi a variante vencedora. Após essa análise o uso de $r = 0{,}20$ e $NI = 64$ foi mantido em todas as demais execuções. O uso de M_j com decaimento exponencial ao longo das iterações embute um tipo de memória importante no PL' porque as metas recentemente estabelecidas têm uma probabilidade maior de serem satisfeitas do que as estabelecida em iterações mais antigas e que possuem pesos menores. Como os pesos M_j afetam a solução do PL', eles influenciam os cálculos de resistência a metas $GR_j(UP)$, $GR_j(DN)$ e os cálculos de penalidade inteira $IP_j(UP)$ e $IP_j(DN)$.

A importância do critério de aspiração por resistência também foi questionada. Assim, foi realizada a execução com o critério de aspiração desabilitado, representada por SemAspir na Tabela 4.2. O teste de Wilcoxon foi inconclusivo, mas o uso da aspiração por resistência falhou em 5 instâncias enquanto que a execução com aspiração desabilitada falhou em 6 instâncias. A execução sem uso de aspiração falhou nas mesmas 5 instâncias em que a execução com aspiração falhou, além da instância acc-6. O desvio médio foi melhor para a aspiração desabilitada cujo valor foi de 89% enquanto que com a aspiração habilitada apresentou o desvio de 93% de ImpRef já discutido no parágrafo anterior. Como um dos objetivos da heurística é alcançar soluções factíveis no maior número possível de instâncias, o critério de aspiração por resistência foi mantido habilitado para os demais testes.

O uso de frações estáticas f_d, g_p e g_s (FracEstática) para determinar a cardinalidade dos conjuntos D_0, G_p e G_s foi comparado com o modelo adaptativo de controle da cardinalidade desses conjuntos adotado em ImpRef. O teste de Wilcoxon mostrou-se inconclusivo. O desvio médio foi melhor na execução FracEstática, 73% contra 93% para ImpRef, porém FracEstática falha em 8 instâncias enquanto ImpRef falha em apenas 5 sendo então considerada a versão vencedora.

A partir dos resultados agregados da Tabela 4.3, observa-se vantagem para ImpRef para desvio

não superior a 5%. Em 60,3% das instâncias, ImpRef esteve abaixo desse valor enquanto M_cte, SemAspir e FracEstática apresentaram respectivamente os valores 43,6%, 56,4% e 56,4%. A implementação ImpRef encontrou 22 (28,2%) soluções ótimas, M_cte encontrou 9 (11,5%), SemAspir 15 (19,2%) e FracEstática 19 (24,4%).

Tab. 4.2: Validação das estratégias da implementação de referência.

Instâncias		ImpRef		M_cte		SemAspir		FracEstática	
Nome	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
10teams	924	**924**	**0**	–	–	**924**	**0**	**924**	**0**
a1c1s1	11503,4	16757,2	46	15633,3	36	16757,2	46	19260,3	67
aflow30a	1158	1227	6	1213	5	1248	8	1294	12
aflow40b	1168	2358	102	1443	24	2325	99	2043	75
air04	56137	56138	0^+	56496	1	56230	0^+	56355	0^+
air05	26374	26444	0^+	26736	1	26475	0^+	26439	0^+
cap6000	-2451377	-2,45E+06	0^+	-2,45E+06	0^+	-2,45E+06	0^+	-2,45E+06	0^+
dano3mip	714,1	743,9	4	731,383	2	743,9	4	725	2
danoint	65,67	67	2	69	5	69,1667	5	68,3333	4
disctom	-5000	–	–	–	–	–	–	–	–
ds	412,503	692,795	68	–	–	692,795	68	–	–
fast0507	174	175	1	177	2	176	1	175	1
fiber	405935	423613	4	519867	28	418662	3	427909	5
fixnet6	3983	3985	0^+	5981	50	3995	0^+	3986	0^+
glass4	1,20E+09	2,70E+09	125	2,05E+09	71	2,40E+09	100	2,28E+09	90
harp2	-7,39E+07	-7,31E+07	1	-7,00E+07	5	-7,16E+07	3	-7,26E+07	2
liu	1496	2096	40	1268	15	1862	24	3150	111
markshare1	1	18	1700	39	3800	18	1700	12	1100
markshare2	1	34	3300	212	21100	34	3300	28	2700
mas74	11801,2	13039,8	10	14098,3	19	13191,6	12	13020	10
mas76	40005,1	40121,9	0^+	42875,8	7	40575,6	1	40321,5	1
misc07	**2810**	**2810**	**0**	**2810**	**0**	**2810**	**0**	**2810**	**0**
mkc	-563,85	-437,49	22	-454,37	19	-480,29	15	-507,112	10
mod011	-5,46E+07	-5,31E+07	3	-5,20E+07	5	-5,41E+07	1	-5,39E+07	1
modglob	2,07E+07	2,08E+07	0^+	2,08E+07	0^+	2,08E+07	0^+	2,09E+07	1
momentum1	109143	–	–	211902	94	–	–	–	–
net12	214	337	57	296	38	337	57	255	19
nsrand-ipx	51200	56640	11	56320	10	56640	11	56160	10
nw04	16862	**16862**	**0**	16968	1	**16862**	**0**	**16862**	**0**
opt1217	**-16**	**-16**	**0**	**-16**	**0**	**-16**	**0**	**-16**	**0**
p2756	3124	3226	3	3176	2	3444	10	3324	6
pk1	11	**11**	**0**	18	64	**11**	**0**	**11**	**0**
pp08a	7350	7810	6	8190	11	7810	6	7830	7
pp08aCUTS	7350	7580	3	7960	8	7640	4	7530	2
protfold	-31	-20	35	–	–	-22	29	-19	39
qiu	-132,873	-36,464	73	**-132,873**	**0**	-124,061	7	**-132,873**	**0**
rd-rplusc-21	165395,3	–	–	–	–	–	–	–	–
set1ch	54537,8	60472,5	11	69906	28	60877	12	60968	12
seymour	423	428	1	424	0^+	428	1	428	1
sp97ar	6,75E+08	7,08E+08	5	7,15E+08	6	7,08E+08	5	7,19E+08	7
stp3d	–	–	–	–	–	–	–	–	–
swath	467,407	522,37	12	589	26	538,319	15	552,161	18
t1717	195779	286071	46	–	–	284486	45	277736	42
tr12-30	130596	166120	27	167073	28	166120	27	164609	26
vpm2	13,75	14,25	4	14	2	14	2	14,25	4
l152lav	4722	**4722**	**0**	4725	0^+	**4722**	**0**	**4722**	**0**
stein45	30	**30**	**0**	**30**	**0**	**30**	**0**	**30**	**0**
ran8x32	5247	5383	3	5450	4	5369	2	5385	3
ran10x26	4270	4373	2	4670	9	4409	3	4392	3
ran12x21	3664	3868	6	4510	23	3846	5	3839	5
ran13x13	3252	**3252**	**0**	3607	11	3334	3	3271	1
binkar10_1	6742,2	7202,81	7	6953,02	3	7328,39	9	7100,29	5
irp	12159,5	**12159,5**	**0**	12162,4	0^+	12159,8	0^+	12159,8	0^+
mas284	91405,7	**91405,7**	**0**	92119,4	1	**91405,7**	**0**	**91405,7**	**0**
prod1	-56	-51	9	-51	9	-51	9	-49	13
bc1	3,33836	3,39703	2	3,3711	1	3,39703	2	3,39687	2
bienst1	46,75	**46,75**	**0**	47,6	2	47,3333	1	**46,75**	**0**
bienst2	54,6	56,5	3	56	3	55,9231	2	56,6667	4
dano3_3	576,345	**576,345**	**0**	**576,345**	**0**	**576,345**	**0**	**576,345**	**0**

Continua na próxima página.

Tab. 4.2 – Continuação da página anterior.

Instâncias		ImpRef		M_cte		SemAspir		FracEstática	
Nome	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
dano3_4	576,435	**576,435**	**0**	**576,435**	**0**	576,499	0^+	**576,435**	**0**
dano3_5	576,925	577,052	0^+	577,052	0^+	577,052	0^+	576,945	0^+
mkc1	-607,15	-604,91	0^+	-606,967	0^+	-606,36	0^+	-606,06	0^+
neos1	19	**19**	**0**	20	5	20	5	**19**	**0**
neos2	454,86	1390,65	206	569,789	25	1259,03	177	1153,61	154
neos3	368,84	1353,98	267	528,78	43	894,611	143	1353,98	267
neos4	-4,86E+10	-4,61E+10	5	-4,53E+10	7	-4,61E+10	5	-4,61E+10	5
neos5	15	**15**	**0**	**15**	**0**	**15**	**0**	**15**	**0**
neos6	83	87	5	93	12	86	4	88	6
seymour1	410,764	411,536	0^+	410,769	0^+	411,536	0^+	411,515	0^+
swath1	379,071	**379,071**	**0**	383,699	1	**379,071**	**0**	**379,071**	**0**
swath2	385,2	**385,2**	**0**	390,869	1	388,05	1	**385,2**	**0**
acc-0	0	**0**	**–**	**0**	**–**	**0**	**–**	**0**	**–**
acc-1	0	**0**	**–**	**0**	**–**	**0**	**–**	**0**	**–**
acc-2	0	**0**	**–**	–	–	**0**	**–**	**0**	**–**
acc-3	0	**0**	**–**	–	–	**0**	**–**	**0**	**–**
acc-4	0	–	–	–	–	–	–	–	–
acc-5	0	1	–	–	–	1	–	–	–
acc-6	0	**0**	–	–	–	–	–	–	–
Média			93		401		89		73
Média(65)			97		406		93		76
Falhas			5		12		6		8

Tab. 4.3: Validação das estratégias da implementação de referência (agregados).

	ImpRef		M_cte		SemAspir		FracEstática	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	32	41	21	26,9	27	34,6	32	41
1% a 5%	15	19,2	13	16,7	17	21,8	12	15,4
5% a 10%	6	7,7	10	12,8	8	10,3	8	10,3
10% a 25%	5	6,4	8	10,3	7	9	7	9
25% a 50%	5	6,4	8	10,3	4	5,1	3	3,8
50% a 100 %	3	3,8	4	5,1	4	5,1	3	3,8
Acima de 100%	6	7,7	2	2,6	4	5,1	5	6,4
Não ótimas em ACCs	1	1,3	0	0	1	1,3	0	0
Falhas	5	6,4	12	15,4	6	7,7	8	10,3

4.1.3 Tratamento da penalidade inteira e da resistência a metas

Para decidir qual é a melhor configuração das componentes da BTP, as três estratégias de cálculo de penalidade inteira e resistência a metas DM, ACVO e RB foram avaliadas. A Tabela 4.4 inclui as técnicas ACVO, DM e RB. O valor da função objetivo e o desvio são apresentados para cada abordagem. A Tabela 4.5 traz os resultados agregados por intervalos de desvio. A ACVO, que é a implementação ImpRef na Tabela 4.2, foi superior à DM e à RB com relação ao teste de Wilcoxon. Entre DM e RB, o teste de Wilcoxon não mostra dominância de uma sobre a outra. O desvio médio foi melhor para a RB, porém a ACVO falhou em apenas 5 instâncias, enquanto que DM e RB falharam

em 9 e 12 instâncias respectivamente. Portanto a ACVO foi a estratégia eleita como a vencedora. Com os resultados agregados da Tabela 4.5, observa-se vantagem para ACVO para desvio não superior a 5%. Em 60,3% das instâncias, ACVO esteve abaixo desse valor enquanto DM e RB apresentaram respectivamente os valores 55,1% e 56,4%. A abordagem ACVO também foi superior no número de soluções ótimas encontradas 22 (28,2%), enquanto DM e RB encontraram 18 (23,1%) e 15 (19,2%) respectivamente.

Tab. 4.4: Estratégias de cálculo da penalidade inteira e da resistência a metas.

Instâncias	ACVO		DM		RB	
Nome	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
10teams	**924**	**0**	**924**	**0**	**924**	**0**
a1c1s1	16757,2	46	16685,8	45	16026,5	39
aflow30a	1227	6	1216	5	1234	7
aflow40b	2358	102	2184	87	2220	90
air04	56138	0^+	56385	0^+	56137	0^+
air05	26444	0^+	26402	0^+	26374	0^+
cap6000	-2,45E+06	0^+	-2,45E+06	0^+	-2,45E+06	0^+
dano3mip	743,9	4	740,876	4	798,588	12
danoint	67	2	65,6667	0^+	70,5	7
disctom	–	–	–	–	–	–
ds	692,795	68	–	–	–	–
fast0507	175	1	175	1	177	2
fiber	423613	4	498313	23	460130	13
fixnet6	3985	0^+	**3983**	**0**	3985	0^+
glass4	2,70E+09	125	2,60E+09	117	4,00E+09	233
harp2	-7,31E+07	1	-7,02E+07	5	-7,09E+07	4
liu	2096	40	1614	8	1892	26
markshare1	18	1700	13	1200	6	500
markshare2	34	3300	26	2500	28	2700
mas74	13039,8	10	12884,9	9	12180,1	3
mas76	40121,9	0^+	40453,4	1	40607,7	2
misc07	**2810**	**0**	**2810**	**0**	**2810**	**0**
mkc	-437,49	22	-402,06	29	-409,56	27
mod011	-5,31E+07	3	-5,22E+07	4	-5,25E+07	4
modglob	2,08E+07	0^+	2,08E+07	0^+	2,08E+07	0^+
momentum1	–	–	–	–	–	–
net12	337	57	–	–	–	–
nsrand-ipx	56640	11	56160	10	54400	6
nw04	**16862**	**0**	**16862**	**0**	16876	0
opt1217	**-16**	**0**	**-16**	**0**	**-16**	**0**
p2756	3226	3	3358	7	3144	1
pk1	**11**	**0**	**11**	**0**	15	36
pp08a	7810	6	7990	9	7820	6
pp08aCUTS	7580	3	7700	5	7670	4
protfold	-20	35	-22	29	–	–
qiu	-36,464	73	-32,0578	76	**-132,873**	**0**
rd-rplusc-21	–	–	–	–	–	–
set1ch	60472,5	11	60499,5	11	62524,5	15
seymour	428	1	427	1	428	1
sp97ar	7,08E+08	5	7,20E+08	7	6,75E+08	0^+
stp3d	–	–	–	–	–	–
swath	522,37	12	554,298	19	544,367	16
t1717	286071	46	332765	70	237217	21
tr12-30	166120	27	167135	28	159582	22
vpm2	14,25	4	14,25	4	14,5	5
l152lav	**4722**	**0**	**4722**	**0**	**4722**	**0**
stein45	**30**	**0**	**30**	**0**	**30**	**0**
ran8x32	5383	3	5422	3	5462	4
ran10x26	4373	2	4410	3	4448	4
ran12x21	3868	6	3997	9	3987	9
ran13x13	**3252**	**0**	3377	4	3393	4
binkar10_1	7202,81	7	6910,12	2	6888,48	2
irp	**12159,5**	**0**	12160,7	0^+	**12159,5**	**0**
mas284	**91405,7**	**0**	**91405,7**	**0**	**91405,7**	**0**
prod1	-51	9	-49	13	-55	2

Continua na próxima página.

Tab. 4.4 – Continuação da página anterior.

Instâncias	ACVO		DM		RB	
Nome	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
bc1	3,39703	2	3,39687	2	3,39804	2
bienst1	**46,75**	**0**	47,25	1	47,25	1
bienst2	56,5	3	56,8333	4	55	1
dano3_3	**576,345**	**0**	**576,345**	**0**	**576,345**	**0**
dano3_4	**576,435**	**0**	**576,435**	**0**	577,37	0^+
dano3_5	577,052	0^+	577,304	0^+	578,907	0^+
mkc1	-604,91	0^+	-603,92	1	-595,36	2
neos1	**19**	**0**	**19**	**0**	20	5
neos2	1390,65	206	1290,55	184	–	–
neos3	1353,98	267	1390,65	277	–	–
neos4	-4,61E+10	5	-4,61E+10	5	-4,61E+10	5
neos5	**15**	**0**	**15**	**0**	**15**	**0**
neos6	87	5	89	7	84	1
seymour1	411,536	0^+	412,646	0^+	411,681	0^+
swath1	**379,071**	**0**	**379,071**	**0**	**379,071**	**0**
swath2	**385,2**	**0**	**385,2**	**0**	388,603	1
acc-0	**0**	–	**0**	–	**0**	–
acc-1	**0**	–	**0**	–	**0**	–
acc-2	**0**	–	**0**	–	**0**	–
acc-3	**0**	–	**0**	–	**0**	–
acc-4	–	–	–	–	–	–
acc-5	1	–	–	–	–	–
acc-6	**0**	–	–	–	–	–
Média		93		74		62
Média(66)		91		70		62
Falhas		5		9		12

Tab. 4.5: Estratégias de cálculo da penalidade inteira e da infactibilidade de metas (agregados).

	ACVO		DM		RB	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	32	41	30	38,5	28	35,9
1% a 5%	15	19,2	13	16,7	16	20,5
5% a 10%	6	7,7	10	12,8	8	10,3
10% a 25%	5	6,4	4	5,1	6	7,7
25% a 50%	5	6,4	4	5,1	4	5,1
50% a 100 %	3	3,8	3	3,8	1	1,3
Acima de 100%	6	7,7	5	6,4	3	3,8
Não ótimas em ACCs	1	1,3	0	0	0	0
Falhas	5	6,4	9	11,5	12	15,4

4.1.4 Resultados computacionais de curto prazo

Finalmente, a BTP foi comparada com as heurísticas do Cplex e com a *Objective Feasibility Pump* (OFP) de Achterberg e Berthold (2007). A exemplo do que Achterberg e Berthold (2007) realizaram, as heurísticas de nó do Cplex 10.0, (HCPLEX), foram executadas com os cortes desabilitados para serem confrontadas com a BTP. Heurísticas de busca local como RINS (Danna et al., 2005) ficaram habilitadas e o melhor resultado encontrado no nó raiz foi registrado. Os resultados da heurística OFP

também foram utilizados nas comparações. A configuração de hardware utilizada por Achterberg e Berthold (2007) é bastante similar à nossa, com uso de um PC com processador Intel Pentium IV de 3.4 GHz e com 3Gbytes de memória RAM. Em todos os cenários, o pré-processamento do Cplex foi aplicado antes das heurísticas serem executadas e o tempo começa a ser computado após o pré-processamento e a resolução do PL do problema original.

A Tabela 4.6 contém os resultados de quatro execuções. Cada uma delas é apresentada em três ou quatro colunas. A coluna **Objetivo** é relativa aos valores de função objetivo, a coluna **Desv** traz o desvio porcentual, a coluna **Tempo** exibe o tempo em segundos em que a melhor solução foi encontrada e a coluna **Num Sol**, quando presente, indica o número de soluções encontradas pela execução. A primeira coluna da tabela é relativa aos nomes das instâncias e quando são registrados em caracteres itálicos indicam que a solução ótima é desconhecida. A linha **Média** no final da tabela traz o desvio médio para as instâncias em que uma solução foi encontrada, a linha **Média(57)** exibe o desvio médio considerando apenas as instâncias em que as 4 execuções conseguiram encontrar uma solução factível e a linha **Falhas** indica em quantas instâncias uma execução falhou.

Há dois resultados da BTP relatados, BTP-ACVO(1^{st}) e BTP-ACVO que diferem apenas no critério de parada. A BTP-ACVO(1^{st}) é relativa à execução da BTP que é interrompida quando a primeira solução inteira-factível é encontrada ou se o limite de tempo de 3600s for atingido. Este critério de parada foi adotado para estabelecer uma base de comparação honesta com a OFP que também pára quando a primeira solução inteira-factível é encontrada. Já a BTP-ACVO é interrompida apenas quando o tempo limite de 3600s é transcorrido e destina-se a avaliar a capacidade da heurística em transcender a primeira solução encontrada quando um tempo computacional razoável é estabelecido. Os resultados HCPLEX são relativos às heurísticas aplicáveis ao nó raiz e o limite de 3600s também foi estabelecido para manter a padronização. Achterberg e Berthold (2007) utilizaram uma versão diferente da instância neos5[2] que estava disponível à época em que nosso trabalho foi realizado. Então, nas comparações que envolvem a OFP, essa instância foi desconsiderada e o resultado relativo a ela aparece com o símbolo '*' na Tabela 4.6. Os resultados em que uma execução alcançou a solução ótima estão destacados em negrito. O símbolo '0^{+}' indica que o desvio ficou abaixo de 0,5%.

O teste de Wilcoxon aplicado à BTP-ACVO(1^{st}) e OFP não estabelece dominância de uma sobre a outra. A BTP-ACVO(1^{st}) vence em 32 instâncias, perde em 37 e há 8 empates, considerando que quando ambas falham ocorre empate. Na média aritmética, a OFP foi melhor com desvio de 922% enquanto que a BTP-ACVO(1^{st}) atingiu 1248%. A OFP falhou em 4 instâncias enquanto a BTP-ACVO(1^{st}) falhou em 5 instâncias. A OFP obteve sucesso nas instâncias disctom, momentum1 e rd-rplusc-21 em que a BTP-ACVO(1^{st}) falhou, enquanto que a BTP-ACVO(1^{st}), por sua vez, conseguiu encontrar soluções para as instâncias ds e acc-6 para as quais a OFP falhou. Nem a OFP e nem a

[2]Baseado em discussões particulares com Achterberg.

HCPLEX obtiveram sucesso com a instância acc-6 e todas as execuções falharam para a instância stp3d. Com relação aos tempos computacionais a BTP-ACVO(1^{st}) é mais rápida em 20 instâncias, a OFP é mais rápida em 28 e há 29 empates.

Na comparação da BTP-ACVO(1^{st}) com a HCPLEX, o teste de Wilcoxon também foi inconclusivo. A BTP-ACVO(1^{st}) venceu em 24 instâncias, perdeu em 46 e houve 8 empates. O desvio médio da HCPLEX foi de 1909% sendo então pior do que o da BTP-ACVO(1^{st}). Com relação à Média(57) a BTP-ACVO(1^{st}) também apresentou resultado melhor. A HCPLEX consome menos tempo computacional, é mais rápida em 41 casos, é mais demorada em 7 casos e há 30 empates, porém falha em 16 instâncias.

O teste de Wilcoxon também foi inconclusivo na comparação da OFP com a HCPLEX. A OFP venceu em 30 instâncias, perdeu em 40 e houve 7 empates. A HCPLEX foi mais rápida em 34, foi mais demorada em 13 e ocorreu empate em 30 casos.

A execução BTP-ACVO foi superior a todos os demais cenários com relação ao teste de Wilcoxon, mostrando a capacidade da BTP de encontrar outras soluções com qualidade melhor quando mais tempo computacional é permitido. Em 60 instâncias, 76,9% dos casos, foi capaz de melhorar a solução encontrada por BTP-ACVO(1^{st}), reduzindo o desvio médio de 1248% para 93%. O tempo médio para a execução BTP-ACVO(1^{st}) encontrar uma solução foi de 340 segundos e o tempo médio da última solução encontrada por BTP-ACVO foi de 1337 segundos. A BTP-ACVO foi melhor que a HCPLEX em 56 instâncias (71,8%) e foi melhor que a OFP em 55 instâncias (70,5%). Respectivamente, houve empates em 10 e 8 casos e a BTP-ACVO perdeu em 12 e 14 instâncias. A BTP-ACVO encontrou a solução ótima para 22 instâncias, representando 28,2% enquanto que para BTP-ACVO(1^{st}), OFP e HCPLEX, os números de soluções ótimas encontradas foram respectivamente 8 (10,2%), 10 (12,9%) e 6 (7,7%). A Tabela 4.7 traz os resultados da Tabela 4.6 agregados por intervalos de desvio. Em 47 instâncias, 60,3%, o desvio esteve abaixo de 5% do ótimo para BTP-ACVO. Para BTP-ACVO(1^{st}), OFP e HCPLEX os números de casos com desvio inferior a 5% foram respectivamente 23 (29,5%), 23 (29,5%) e 28 (35,9%).

Tab. 4.6: Resultados de curto prazo.

	BTP-ACVO(1^{st})			BTP-ACVO				Objective Feasibility Pump			HCPLEX		
Nome	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo	Num Sol	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo
10teams	1054	14	28	**924**	**0**	**346**	**13**	952	3	5	–	–	1
a1c1s1	19328,2	68	17	16757,2	46	20	21	16076,6	40	1	21029,4	83	1
aflow30a	3100	168	1	1227	6	2031	33	4105	254	0	1239	7	0
aflow40b	6125	424	17	2358	102	1351	21	2049	75	0	1439	23	1
air04	56496	1	14	56138	0^+	733	6	57298	2	164	–	–	6
air05	27485	4	8	26444	0^+	1805	16	26942	2	8	27291	3	3
cap6000	-2,45E+06	0^+	37	-2,45E+06	0^+	37	1	-2,43E+06	1	0	-2,45E+06	0^+	0
dano3mip	856,5	20	117	743,9	4	315	4	769,3	8	383	714,125	0	140
danoint	82	25	0	67	2	2209	3	87	32	3	–	–	1
disctom	–	–	3600	–	–	3600	0	**-5000**	**0**	**11**	–	–	6
ds	701,055	70	3313	692,795	68	3315	2	–	100	3600	412,5025	0	88
fast0507	183	5	36	175	1	981	9	179	3	21	177	2	32
fiber	567919	40	0	423613	4	1454	9	1,21E+06	197	0	468924	16	0
fixnet6	5376	35	0	3985	0^+	2594	29	4807	21	0	4435	11	0

Continua na próxima página.

Tab. 4.6 – Continuação da página anterior.

	BTP-ACVO(1^{st})			BTP-ACVO				Objective Feasibility Pump			HCPLEX		
Nome	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo	Num Sol	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo
glass4	3,90E+09	225	0	2,70E+09	125	3	4	3,10E+09	158	0	–	–	0
harp2	-6,42E+07	13	5	-7,31E+07	1	1366	5	-5,59E+07	24	0	-7,26E+07	2	0
liu	4292	187	0	2096	40	246	40	4100	174	1	4674	212	1
markshare1	234	23300	0	18	1700	680	17	194	19300	1	230	22900	0
markshare2	553	55200	0	34	3300	2547	20	365	36400	0	898	89700	0
mas74	16039,8	36	0	13039,8	10	2135	15	19033,1	61	0	14372,9	22	0
mas76	42875,8	7	0	40121,9	0^+	3320	14	50124	25	0	**40005,1**	**0**	**0**
misc07	3745	33	0	**2810**	**0**	**1**	**8**	3425	22	0	**2810**	**0**	**0**
mkc	-437,49	22	28	-437,49	22	28	1	-289,95	49	0	-528,53	6	1
mod011	-5,17E+07	5	0	-5,31E+07	3	3132	3	-4,56E+07	16	1	-4,74E+07	13	0
modglob	3,62E+07	74	0	2,08E+07	0^+	302	80	2,11E+07	2	0	2,08E+07	0^+	0
momentum1	–	–	3600	–	–	3600	0	346535	218	223	527461	383	21
net12	337	57	69	337	57	69	1	337	57	14	–	–	22
nsrand-ipx	57920	13	1	56640	11	10	3	89120	74	1	55680	9	1
nw04	17514	4	4	**16862**	**0**	**1747**	**5**	17856	6	9	16956	1	1
opt1217	**-16**	**0**	**0**	**-16**	**0**	**0**	**1**	**-16**	**0**	**0**	**-16**	**0**	**0**
p2756	59846	1816	5	3226	3	79	12	89266	2757	3	3425	10	0
pk1	31	182	0	**11**	**0**	**3264**	**15**	83	655	0	18	64	0
pp08a	12670	72	0	7810	6	2931	27	10940	49	0	8120	10	0
pp08aCUTS	10350	41	0	7580	3	909	33	8530	16	0	8100	10	0
protfold	-14	55	40	-20	35	1085	5	-12	61	268	-20	35	15
qiu	318,857	340	1	-36,464	73	2572	15	625,709	571	0	173,979	231	1
rd-rplusc-21	–	–	3600	–	–	3600	0	171182	3	790	–	–	23
set1ch	84281	55	0	60472,5	11	1516	42	84167,5	54	0	66772,5	22	0
seymour	438	4	1	428	1	4	11	445	5	3	434	3	11
sp97ar	7,11E+08	5	11	7,08E+08	5	23	2	9,41E+08	39	3	6,83E+08	1	13
stp3d	–	–	3600	–	–	3600	0	–	–	3600	–	–	3016
swath	711,687	52	5	522,37	12	3462	13	1280,95	174	13	1521,66	226	0
t1717	351388	79	475	286071	46	3449	17	195779	0	171	340588	74	28
tr12-30	253639	94	3	166120	27	5	95	163794	25	0	–	–	0
vpm2	17,25	25	0	14,25	4	3509	15	15,25	11	0	15,25	11	0
l152lav	4770	1	1	**4722**	**0**	**646**	**9**	4757	1	0	4760	1	0
stein45	**30**	**0**	**0**	**30**	**0**	**0**	**1**	35	17	0	**30**	**0**	**0**
ran8x32	6047	15	0	5383	3	3011	24	5817	11	0	5837	11	0
ran10x26	5101	19	0	4373	2	3341	12	4833	13	0	4745	11	0
ran12x21	4597	25	0	3868	6	3378	22	4231	15	0	4080	11	0
ran13x13	4267	31	0	**3252**	**0**	**1228**	**29**	3820	17	0	3508	8	0
binkar10_1	7875,64	17	1	7202,81	7	94	3	7156,21	6	1	6917,17	3	0
irp	12310,3	1	1	**12159,5**	**0**	**405**	**4**	12162,4	0^+	1	12162,4	0^+	0
mas284	94429,6	3	0	**91405,7**	**0**	**166**	**12**	99522,7	9	0	93708,1	3	0
prod1	-41	27	0	-51	9	161	10	-53	5	0	-49	13	0
bc1	3,43703	3	1	3,39703	2	6	3	5,4391	63	2	3,44084	3	0
bienst1	59,67	28	0	**46,75**	**0**	**9**	**8**	55,5	19	0	66,5	42	0
bienst2	70,7	29	0	56,5	3	162	5	73,6667	35	1	62,1667	14	0
dano3_3	**576,345**	**0**	**84**	**576,345**	**0**	**84**	**1**	**576,345**	**0**	**47**	576,396	0^+	69
dano3_4	576,499	0^+	13	**576,435**	**0**	**1599**	**2**	**576,435**	**0**	**76**	576,499	0^+	66
dano3_5	578,7	0^+	388	577,1	0^+	439	2	576,994	0^+	106	578,648	0^+	91
mkc1	-595,49	2	12	-604,91	0^+	2432	7	-563,1	7	0	-604,86	0^+	1
neos1	24	26	2	**19**	**0**	**5**	**6**	68	258	1	21	11	0
neos2	1390,65	206	7	1390,65	206	7	1	958,977	111	7	–	–	0
neos3	1390,65	277	124	1353,98	267	124	2	1630,21	342	8	–	–	0
neos4	-4,53E+10	7	1	-4,61E+10	5	3600	247	-4,81E+10	1	3	-4,83E+10	1	1
neos5	16	7	0	**15**	**0**	**1**	**4**	*	*	*	15,5	3	0
neos6	94	13	32	87	5	1643	5	93	12	12	91	10	4
seymour1	412,052	0^+	2	411,536	0^+	9	3	427,063	4	3	412,448	0^+	10
swath1	382,308	1	1	**379,071**	**0**	**2306**	**4**	439,106	16	2	879,034	132	0
swath2	396,016	3	1	**385,2**	**0**	**2193**	**5**	641,544	67	2	1028,07	167	0
acc-0	**0**	–	**1**	**0**	–	**1**	**1**	**0**	–	**0**	**0**	–	**0**
acc-1	**0**	–	**3**	**0**	–	**3**	**1**	**0**	–	**1**	**0**	–	**3**
acc-2	**0**	–	**57**	**0**	–	**57**	**1**	**0**	–	**3**	–	–	8
acc-3	**0**	–	**128**	**0**	–	**128**	**1**	**0**	–	**12**	–	–	14
acc-4	–	–	3600	–	–	3600	0	–	–	3600	–	–	14
acc-5	1	–	649	1	–	649	1	**0**	–	**2054**	–	–	7
acc-6	**0**	–	**2795**	**0**	–	**2795**	**1**	–	–	3600	–	–	9
Média		1248	340		93	1337	14		922	245		1909	48
Média(57)		1450			96				1083			2003	
Falhas		5			5				4			16	

Tab. 4.7: Resultados agregados de curto prazo.

	BTP-ACVO (1^{st})		BTP-ACVO		OFP		HCPLEX	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	14	17,9	32	41	16	20,5	19	24,4
1% a 5%	9	11,5	15	19,2	7	9	9	11,5
5% a 10%	6	7,7	6	7,7	7	9	6	7,7
10% a 25%	10	12,8	5	6,4	14	17,9	15	19,2
25% a 50%	12	15,4	5	6,4	8	10,3	2	2,6
50% a 100 %	10	12,8	3	3,8	8	10,3	3	3,8
Acima de 100%	11	14,1	6	7,7	14	17,9	8	10,3
Não ótimas em ACCs	1	1,3	1	1,3	0	0	0	0
Falhas	5	6,4	5	6,4	4	5,1	16	20,5

4.2 Resultados computacionais de extensões do curto prazo

A apresentação dos resultados das extensões foi dividida em duas partes. Como a nossa implementação do *probing* se aplica apenas a instâncias de programação inteira pura, foi necessário compor tabelas separadas para a sua discussão. A técnica de fixação de variáveis pela análise de custo reduzido não teve impacto e então foi suprimida. O uso de *branch-and-cut* para resolver infactibilidade de metas, o uso de cortes e a relaxação da restrição de função objetivo puderam ser analisados simultaneamente.

4.2.1 Calibração dos parâmetros das extensões

A abordagem de *probing* não tem parâmetros a serem calibrados e em todas as iterações o procedimento é invocado. Na relaxação da restrição de função objetivo da expressão 3.1, adotou-se $\Delta ss = 50$, *MaxRelax* = 0,03 e $\Delta rm = 500$. Aumentar *MaxRelax* além do valor escolhido leva a busca a reencontrar soluções previamente visitadas com muita freqüência. Diminuir Δss abaixo do valor escolhido provoca ciclagem. O valor alto de Δrm favorece o afastamento da região onde uma solução inteira foi encontrada, pois o aumento na relaxação ocorre de maneira bastante suave.

Com relação ao uso de *branch-and-cut* para resolver infactibilidade de metas, dependendo de como os parâmetros do algoritmo da Figura 3.2 são escolhidos, o seu comportamento muda. Por exemplo, se um tempo grande é oferecido para o *branch-and-cut* e se a fração de variáveis com metas atendidas e que são tratadas como potencialmente infactíveis for muito elevada, a BTP degenera para um modelo misto envolvendo heurística com *branch-and-cut*. Como a proposta única desta tese é avaliar a BTP e as suas componentes internas, os parâmetros do algoritmo da Figura 3.2 foram

testados apenas para valores relativamente baixos. A idéia foi simplesmente avaliar até que ponto a abordagem adaptativa de controle da cardinalidade dos conjuntos G_p, G_s e D_0 é eficaz. Com isso, procurou-se verificar se, com tempo reduzido, o *branch-and-cut* conseguiria resolver, sem uso do conjunto G_s, uma infactibilidade de metas que se mostrou intratável pela abordagem adaptativa de controle da cardinalidade dos conjuntos. Assim foram adotados **FracPermitida** = 0,2, **FracPorIter** = 0,1, **TempoRConf** = 10s e **multRR** = 3,0. Com essa configuração, até 20% das variáveis com metas atendidas podem ser tratadas como potencialmente infactíveis. Além disso, durante até 10 iterações do algoritmo da Figura 3.2 o *branch-and-cut* é ativado com limite de 10 segundos. Resultados de qualidade inferior foram observados ao limitar o *branch-and-cut* em 5 ou 20 segundos.

O uso de cortes foi calibrado com **MaxCortes** $\in \{20, 40, 80\}$ e **MinDuracaoCortes** $\in \{20, 40, 80\}$. Os cortes foram testados com e sem uso de memória tabu e a melhor configuração encontrada foi **MaxCortes** = 40, **MinDuracaoCortes** = 40 e com a memória tabu desativada.

4.2.2 Resultados das extensões

As Tabelas 4.8 e 4.9 contemplam os resultados da execução com *probing* ativado, BTP-ACVO (*probing*), confrontados com os da execução de referência BTP-ACVO. As duas primeiras colunas da Tabela 4.8 são referentes aos nomes das instâncias e aos melhores valores conhecidos de função objetivo. Na seqüência, cada uma das duas execuções é apresentada em duas colunas, a primeira referente ao valor de função objetivo alcançado e a segunda relativa ao desvio em relação à melhor solução conhecida. No teste de Wilcoxon, não foi constatada uma dominância. Das 25 instâncias de programação inteira pura, a BTP-ACVO foi melhor em 9, houve 12 empates e a BTP-ACVO (*probing*) foi superior em 4 instâncias. Destaca-se no uso de *probing* a descoberta de soluções ótimas para as instâncias disctom e acc-4. Nenhuma outra variante da BTP foi capaz de encontrar uma solução válida para a instância disctom. Por outro lado, com o uso do *probing* ocorrem 4 falhas enquanto a BTP-ACVO falhou em 3 instâncias. As médias não demonstraram disparidade, ambas execuções apresentaram desvio médio de 10%. Quando a média é considerada apenas para as 15 instâncias em que ambas execuções encontraram pelo menos uma solução inteira, a BTP-ACVO atingiu desvio de 6% e a BTP-ACVO (*probing*) subiu para 11%. Da Tabela 4.9 com desvios agregados, observa-se que as execuções foram equilibradas. Ambas execuções encontraram 9 (36%) soluções ótimas. Para desvio não superior a 5%, a BTP-ACVO e BTP-ACVO (*probing*) foram competitivas, atingindo essa faixa para respectivamente 72% e 68% das instâncias.

Tab. 4.8: Resultados do *Probing*.

Instâncias		BTP-ACVO		BTP-ACVO (*Probing*)	
Nome	Melhor Valor	Objetivo	Desv (%)	Objetivo	Desv (%)
air04	56137	56138	0	56139	0
air05	26374	26444	0	26453	0
cap6000	-2,45E+06	-2,45E+06	0	-2,45E+06	0
disctom	-5000	–	–	**-5000**	**0**
ds	412,5025	692,795	68	–	–
fast0507	174	175	1	177	2
harp2	-7,39E+07	-7,31E+07	1	-7,29E+07	1
nw04	16862	**16862**	**0**	**16862**	**0**
p2756	3124	3226	3	3226	3
protfold	-31	-20	35	-22	29
seymour	423	428	1	426	1
sp97ar	6,75E+08	7,08E+08	5	7,11E+08	5
stp3d	–	–		–	–
t1717	195779	286071	46	286071	46
l152lav	4722	**4722**	**0**	**4722**	**0**
stein45	30	**30**	**0**	**30**	**0**
ran8x32	5247	5383	3	5383	3
neos1	19	**19**	**0**	33	74
acc-0	0	**0**	–	**0**	–
acc-1	0	**0**	–	**0**	–
acc-2	0	**0**	–	**0**	–
acc-3	0	**0**	–	**0**	–
acc-4	0	–	–	**0**	–
acc-5	0	1	–	–	–
acc-6	0	**0**	–	–	–
Média			10		10
Média(15)			6		11
Falhas			3		4

Os demais resultados das extensões constam das Tabelas 4.10 e 4.11 e são comparados com a implementação de referência BTP-ACVO. A execução BTP-Cortes é relativa ao uso de cortes, BTP-IM-B&C é relativa ao uso de *branch-and-cut* para resolver infactibilidade de metas e BTP-RFO representa o uso da relaxação na restrição de função objetivo. A inclusão de cortes foi facilitada devido ao uso do PL de duas fases, uma vez que a solução ótima da fase I oferece automaticamente um

Tab. 4.9: Resultados agregados de *probing*.

	BTP-ACVO		BTP-ACVO (*Probing*)	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	13	52	13	52
1% a 5%	5	20	4	16
5% a 10%	0	0	1	4
10% a 25%	1	4	0	0
25% a 50%	2	8	2	8
50% a 100 %	0	0	1	4
Acima de 100%	0	0	0	0
Não ótimas em ACCs	1	4	0	0
Falha	3	12	4	16

corte, conforme a discussão da Subseção 3.1.4. A execução BTP-Cortes teve o mérito de encontrar a solução ótima para a instância acc-5 e também foi capaz de encontrar solução factível para a instância momentum1. A implementação do uso de cortes é menos trabalhosa do que a da memória tabu que foi usada, porém os resultados favorecem a manutenção da memória tabu. Com relação aos desvios médios, a BTP-Cortes foi ligeiramente inferior à BTP-ACVO. A BTP-ACVO alcançou média de 93% enquanto BTP-Cortes alcançou 96%. Quando Média(63) é considerada, a vantagem para a BTP-ACVO se mantém. O teste de Wilcoxon mostrou-se inconclusivo, a BTP-ACVO leva vantagem em 30 casos, perde em 23 e ocorrem 25 empates. Mas o uso de cortes falhou em 7 instâncias, e então foi desabilitado. De acordo com a Tabela 4.11, ambas execuções alcançaram desvio inferior a 5% para 47 instâncias, o que equivale a 60,3% do total.

O uso de *branch-and-cut* para resolver infactibilidade de metas não foi superior ao uso da abordagem adaptativa de controle dos conjuntos G_p, G_s e D_0. O teste de Wilcoxon foi inconclusivo, a BTP-IM-B&C foi melhor em 26 instâncias, a BTP-ACVO foi melhor em 18 e houve 34 empates. A execução BTP-IM-B&C fracassou em encontrar uma solução inteira-factível em 9 instâncias enquanto BTP-ACVO falhou em 5. A BTP-IM-B&C atinge médias de desvio melhores, 84% e 82% e a BTP atinge 93% e 90% respectivamente para Média e Média(63). A execução BTP-IM-B&C encontra solução para a instância momentum1, uma tarefa que não se mostrou fácil para a busca tabu paramétrica. Quando a Tabela 4.11 é usada para comparar as duas abordagens, a BTP-IM-B&C novamente não se mostra muito promissora. Em 53,8% das instâncias, o desvio ficou abaixo de 5%, enquanto que a BTP-ACVO atingem 60,3% das instâncias para esse mesmo desvio. A execução BTP-ACVO realizou em média 91,4% a mais de iterações do que a BTP-IM-B&C para o tempo limite de 3600 segundos. Dedicar menos esforço computacional por cada iteração em que a busca se encontra na infactibilidade de metas com a contrapartida de executar mais iterações com tempo limite de uma

hora mostrou-se ser mais vantajoso.

A relaxação da restrição de função objetivo não teve influência suficiente para encontrar soluções factíveis nas instâncias em que a BTP-ACVO falhou. O teste de Wilcoxon mostrou-se inconclusivo. A BTP-ACVO foi melhor em 23 instâncias, a BTP-RFO foi melhor em 17 e houve 38 empates. A BTP-RFO foi capaz de reduzir o desvio médio, levando de 93% para 82% e de 90% para 78% quando Média(63) é considerada. Porém quando olhado à luz dos dados agregados na Tabela 4.11, essa vantagem se mostra menos significativa. A implementação BTP-ACVO alcançou desvio inferior a 5% em 60,3% das instâncias enquanto que a BTP-RFO alcançou essa marca para 56,4% dos casos. A BTP-RFO encontrou 19 (24,4%) soluções ótimas, a BTP-Cortes encontrou 18 (23,1%) e a BTP-IM-B&C 16 (20,5%).

Tab. 4.10: Resultados das demais extensões.

Instância	BTP-ACVO		BTP-Cortes		BTP-IM-B&C		BTP-RFO	
Nome	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
10teams	**924**	**0**	**924**	**0**	**924**	**0**	**924**	**0**
a1c1s1	16757,2	46	17009,1	48	17466,6	52	16757,2	46
aflow30a	1227	6	1303	13	1431	24	1267	9
aflow40b	2358	102	2076	78	2598	122	2346	101
air04	56138	0^+	56167	0^+	56151	0^+	56238	0^+
air05	26444	0^+	26444	0^+	26470	0^+	26490	0^+
cap6000	-2,45E+06	0^+	-2,45E+06	0^+	-2,45E+06	0^+	-2,45E+06	0^+
dano3mip	743,9	4	757,1	6	757,1	6	743,9	4
danoint	67	2	68	4	69	5	68,75	5
disctom	–	–	–	–	–	–	–	–
ds	692,795	68	–	–	–	–	692,795	68
fast0507	175	1	175	1	176	1	175	1
fiber	423613	4	437256	8	436432	8	452654	12
fixnet6	3985	0^+	**3983**	**0**	3985	0^+	4014	1
glass4	2,70E+09	125	2,28E+09	90	2,33E+09	94	2,42E+09	101
harp2	-7,31E+07	1	-7,28E+07	2	-7,14E+07	3	-7,23E+07	2
liu	2096	40	1956	31	1780	19	1760	18
markshare1	18	1700	18	1700	10	900	9	800
markshare2	34	3300	38	3700	37	3600	36	3500
mas74	13039,8	10	12919,4	9	13622,8	15	13354,4	13
mas76	40121,9	0^+	40595,6	1	40657,1	2	40913,3	2
misc07	**2810**	**0**	**2810**	**0**	**2810**	**0**	**2810**	**0**
mkc	-437,49	22	-463,38	18	-447,05	21	-452,05	20
mod011	-5,31E+07	3	-5,39E+07	1	-5,40E+07	1	-5,36E+07	2
modglob	2,08E+07	0^+	2,08E+07	0^+	2,08E+07	0^+	2,08E+07	0^+
momentum1	–	–	218242	100	179763,6	65	–	–
net12	337	57	–	–	–	–	337	57
nsrand-ipx	56640	11	56480	10	56640	11	56640	11
nw04	**16862**	**0**	**16862**	**0**	**16862**	**0**	**16862**	**0**
opt1217	**-16**	**0**	**-16**	**0**	**-16**	**0**	**-16**	**0**
p2756	3226	3	3244	4	3293	5	3278	5
pk1	**11**	**0**	13	18	18	64	15	36
pp08a	7810	6	8010	9	8000	9	7850	7
pp08aCUTS	7580	3	7710	5	7620	4	7530	2
protfold	-20	35	-23	26	-22	29	-23	26
qiu	-36,464	73	**-132,873**	**0**	**-132,873**	**0**	-115,248	13
rd-rplusc-21	–	–	–	–	–	–	–	–
set1ch	60472,5	11	59852	10	59517	9	61772,5	13
seymour	428	1	426	1	425	0^+	428	1
sp97ar	7,08E+08	5	7,02E+08	4	7,00E+08	4	6,94E+08	3
stp3d	–	–	–	–	–	–	–	–
swath	522,37	12	528,451	13	535,206	15	532,146	14
t1717	286071	46	358208	83	293964	50	314376	61
tr12-30	166120	27	163338	25	161341	24	166120	27
vpm2	14,25	4	14,25	4	14,00	2	14,50	5
l152lav	**4722**	**0**	**4722**	**0**	**4722**	**0**	**4722**	**0**
stein45	**30**	**0**	**30**	**0**	**30**	**0**	**30**	**0**

Continua na próxima página.

Tab. 4.10 – Continuação da página anterior.

Instância	BTP-ACVO		BTP-Cortes		BTP-IM-B&C		BTP-RFO	
Nome	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
ran8x32	5383	3	5454	4	5504	5	5439	4
ran10x26	4373	2	4456	4	4427	4	4468	5
ran12x21	3868	6	3999	9	3998	9	3853	5
ran13x13	**3252**	**0**	3371	4	3418	5	3398	4
binkar10_1	7202,81	7	6838,44	1	6878,53	2	7207,16	7
irp	**12159,5**	**0**	12160,2	0^+	**12159,5**	**0**	**12159,5**	**0**
mas284	**91405,7**	**0**	**91405,7**	**0**	**91405,7**	**0**	**91405,7**	**0**
prod1	-51	9	-49	13	-49	13	-51	9
bc1	3,39703	2	3,39703	2	3,39703	2	3,39687	2
bienst1	**46,75**	**0**	47,00	1	47,00	1	**46,75**	**0**
bienst2	56,5	3	55,4	1	55,0	1	55,7	2
dano3_3	**576,345**	**0**	**576,345**	**0**	**576,345**	**0**	**576,345**	**0**
dano3_4	**576,435**	**0**	576,499	0^+	576,499	0^+	**576,435**	**0**
dano3_5	577,052	0^+	577,052	0^+	577,052	0^+	577,052	0^+
mkc1	-604,91	0^+	-604,49	0^+	-605,43	0^+	-603,91	1
neos1	**19**	**0**	**19**	**0**	**19**	**0**	**19**	**0**
neos2	1390,65	206	1325,90	191	–	–	1353,98	198
neos3	1353,98	267	–	–	1333,73	262	1353,98	267
neos4	-4,61E+10	5	-4,61E+10	5	-4,61E+10	5	-4,61E+10	5
neos5	**15**	**0**	**15**	**0**	**15**	**0**	**15**	**0**
neos6	87	5	84	1	87	5	87	5
seymour1	411,536	0^+	412,052	0^+	412,052	0^+	411,536	0^+
swath1	**379,071**	**0**	382,308	1	382,308	1	**379,071**	**0**
swath2	**385,2**	**0**	385,973	0^+	387,829	1	387,650	1
acc-0	**0**	**–**	**0**	**–**	**0**	**–**	**0**	**–**
acc-1	**0**	**–**	**0**	**–**	**0**	**–**	**0**	**–**
acc-2	**0**	**–**	**0**	**–**	**0**	**–**	**0**	**–**
acc-3	**0**	**–**	**0**	**–**	**0**	**–**	**0**	**–**
acc-4	–	–	–	–	–	–	**–**	–
acc-5	1	–	**0**	**–**	–	–	1	–
acc-6	**0**	**–**	**0**	**–**	–	–	**0**	**–**
Média		93		96		84		82
Média(63)		90		95		82		78
Falhas		5		7		9		5

Tab. 4.11: Resultados agregados das demais extensões.

	BTP-ACVO		BTP-Cortes		BTP-IM-B&C		BTP-RFO	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	32	41	32	41	30	38,5	29	37,2
1% a 5%	15	19,2	15	19,2	12	15,4	15	19,2
5% a 10%	6	7,7	7	9	9	11,5	7	9
10% a 25%	5	6,4	6	7,7	8	10,3	8	10,3
25% a 50%	5	6,4	4	5,1	1	1,3	4	5,1
50% a 100 %	3	3,8	4	5,1	5	6,4	3	3,8
Acima de 100%	6	7,7	3	3,8	4	5,1	6	7,7
Não ótimas em ACCs	1	1,3	0	0	0	0	1	1,3
Falhas	5	6,4	7	9,0	9	11,5	5	6,4

4.3 Resultados com uso de memória de longo prazo

A implementação de referência foi a adotada para a inclusão dos módulos baseados em longo prazo. Assim, a técnica ACVO foi a escolhida para resolver a infactibilidade de metas e a infactibilidade inteira. O uso de M_j dinâmico foi mantido, bem como o critério de aspiração baseado em resistência. Foram avaliados o uso da matriz de freqüência, geração de centros de intensificação e diversificação e o uso de PIM residual após a fixação de variáveis consistentes e fortemente determinadas.

4.3.1 Calibração dos parâmetros de longo prazo

A calibração dos parâmetros das estratégias de longo prazo foi dividida em calibração não acurada e calibração fina. A calibração não acurada foi conduzida por meio de 26 das 78 instâncias da Miplib que apresentam os maiores desvios em relação aos melhores valores conhecidos de função objetivo ou instâncias em que a BTP falhou com maior freqüência para diferentes abordagens de curto prazo. O tempo limite de 20 minutos foi utilizado para fazer essa calibração. Depois, para as melhores configurações de parâmetros, a calibração fina com limite de 40 minutos e envolvendo todas as 78 instâncias foi aplicada.

Há muitos parâmetros interdependentes cuja calibração simultânea é computacionalmente proibitiva. Para contornar essa dificuldade, alguns parâmetros receberam valores iniciais que apresentaram resultados razoáveis, embora não fossem ainda completamente ajustados. Para essa finalidade, foram usados testes preliminares e estatística. A cardinalidade b do conjunto de referência R, após a aplicação de valores tentativos preliminares foi mantida, antes de ser efetivamente calibrada, em $b = 21$. Os parâmetros de controle de duração das fases de curto e longo prazos foram também preliminarmente ajustados em $\alpha\_cc = 0{,}71$, $\alpha\_int = 0{,}35$ e $\alpha\_div = 0{,}35$. Esses valores iniciais ainda não calibrados foram determinados a partir de alguns dados estatísticos da execução de referência de curto prazo. Para todas as instâncias em que uma solução inteira foi encontrada, foi calculada a relação entre o número de iterações dispendidas para encontrar a primeira solução inteira-factível e o número de variáveis binárias n. A média desse valor, considerando todas essas instâncias, acrescida do desvio padrão resultou em 0,71, e que foi o valor inicial usado para α_cc. Para a duração das fases de longo prazo, foi inicialmente adotada a metade desse valor.

Iniciado o processo de calibração de longo prazo, foram tratados os parâmetros que determinam as durações das fases de curto e longo prazos quando é usada a matriz de freqüência e os que controlam a inclusão dos elementos no conjunto de referência. Calibrá-los simultaneamente seria computacionalmente muito dispendioso, então a solução encontrada foi calibrá-los em três etapas, sendo as duas primeiras não acuradas e a terceira de ajuste fino, de acordo com a Tabela

Tab. 4.12: Etapas da calibração dos parâmetros ξ, ω, α_cc, α_int e α_div.

Etapa	Tipo	Descrição
1	Não acurada	Matriz de freqüência ativa, com α_cc, α_int, α_div e b fixos em valores preliminares. Testar pares $\xi \in \{0{,}1; 0{,}2; \ldots; 0{,}9; 1\}$ e $\omega \in \{1{,}1; 1{,}2; 1{,}3; 1{,}4\}$. Guadar os 3 melhores pares.
2	Não acurada	Usar o melhor par da Etapa 1 como referência. Testar pares $\alpha\_cc = 0{,}71 + \Delta_I$, tal que $\Delta_I \in \{0; 0{,}25; 0{,}50; \ldots; 2{,}00; 2{,}25; 2{,}50\}$ e $\alpha\_int = \alpha\_div = 0{,}35 + \Delta_{II}$ tal que $\Delta_{II} \in \{-0{,}2; -0{,}1; 0; 0{,}1; 0{,}2\}$. Guardar os 4 melhores pares.
3	Fina	Combinar os 3 melhores pares de resultados da Etapa 1 com os 4 melhores pares da Etapa 2 e escolher o melhor deles.

4.12. Na primeira etapa, foram calibrados os parâmetros que controlam a inclusão dos elementos no conjunto de referência. A matriz de freqüência ficou ativa com os parâmetros fixos nos valores preliminares $\alpha\_cc = 0{,}71$, $\alpha\_int = 0{,}35$ e $\alpha\_div = 0{,}35$. Além disso, a cardinalidade do conjunto R foi mantida em $b = 21$. O parâmetro ξ da expressão (3.6) e ω que estabelece um limiar de qualidade discutidos na Subseção 3.2.1 foram simultaneamente testados para pares de valores $\xi \in \{0{,}1; 0{,}2; \ldots; 0{,}9; 1\}$ e $\omega \in \{1{,}1; 1{,}2; 1{,}3; 1{,}4\}$. Os três melhores pares de valores para ξ e ω foram guardados para serem usados na calibração fina da etapa 3 e o melhor par serviu de referência para a etapa 2. A segunda etapa do processo envolveu calibrar os parâmetros α_cc, α_int e α_div. Para impedir um número muito grande de combinações de parâmetros, adotou-se $\alpha\_int = \alpha\_div$, reduzindo assim para dois o número de parâmetros a serem calibrados. Os valores testados foram $\alpha\_cc = 0{,}71 + \Delta_I$, tal que $\Delta_I \in \{0; 0{,}25; 0{,}50; \ldots; 2{,}00; 2{,}25; 2{,}50\}$ e $\alpha\_int = \alpha\_div = 0{,}35 + \Delta_{II}$ tal que $\Delta_{II} \in \{-0{,}2; -0{,}1; 0; 0{,}1; 0{,}2\}$ e os quatro melhores valores foram registrados. A terceira etapa envolveu combinar os três melhores resultados da etapa 1 com os quatro melhores resultados da etapa 2 e aplicar a execução por 40 minutos para todas as 78 instâncias. Dessas doze combinações, os melhores valores encontrados foram $\xi = 0{,}7$, $\omega = 1{,}3$, $\alpha\_cc = 0{,}71 + 1{,}75$ e $\alpha\_int = \alpha\_div = 0{,}35 - 0{,}2$. Variações de $\pm 0{,}1$ em α_int não produziram resultados melhores.

A cardinalidade b do conjunto de referência R e o número k de elementos escolhidos para compor um centro $R(k)$ foram então calibrados considerando a geração de centros de intensificação e diversificação para o algoritmo *RK*_1 da Figura 3.7 com o uso da matriz de freqüência ativado. Para a calibração não acurada, foram testados pares de valores b e k tais que $b \in \{17, 21, 25, 29, 33, 37\}$ e $k \in \{3, 5, 7\}$. As três melhores configurações foram reexecutadas usando a calibração fina e a melhor configuração $b = 29$ e $k = 3$ foi encontrada. Na seqüência, foram calibrados os parâmetros V_{min}, V_{max} da expressão (3.22) e f da expressão (3.21) e que fazem parte da geração de centros com uso do algoritmo *RK*_2 da Figura 3.8. Os parâmetros V_{min} e V_{max} foram testados para $V_{min} \in \{2, 3, 4, 5, 6\}$

Tab. 4.13: Calibração dos parâmetros dos algoritmos FixaVars e Coordenador.

Etapa	Tipo	Descrição
1	Não acurada	Manter **fracaoLimite**, **nivelMax** e **maxRk** em valores preliminares. Manter $\gamma_2 = 50$. Testar **maxAnalisar** $\in \{2\%; 2{,}5\%; 3\%; 3{,}5\%; 4\%\}$ e **maxFix** $\in \{0{,}5\%; 1\%; 1{,}5\%; 2\%\}$. Guadar os 3 melhores pares.
2	Não acurada	Usar o melhor par da Etapa 1 como referência. Manter **fracaoLimite** ainda no valor preliminar. Testar pares **maxRk** $\in \{2, 3, 4\}$ e **nivelMax** $\in \{2, 3, 4\}$. Guardar os 3 melhores pares.
3	Fina	Combinar os 3 melhores pares de resultados da Etapa 1 com os 3 melhores pares da Etapa 2.
4	Fina	Testar valores **fracaoLimite** $\in \{50\%, 70\%, 80\%, 90\%, 95\%, 99\%\}$ com o melhor conjunto de parâmetros determinados na Etapa 3.

e $V_{max} \in \{6, 7, 8, 9, 10\}$, mantendo f fixo em $f = 0{,}5$ considerando calibração não acurada. Os três melhores resultados foram coletados e aplicados na calibração fina com $f \in \{0{,}2; 0{,}3; 0{,}5; 0{,}7\}$. A melhor configuração encontrada é formada por $V_{min} = 2$, $V_{max} = 8$ e $f = 0{,}3$.

O último conjunto de parâmetros calibrados foi o da abordagem baseada em fixação de variáveis fortemente determinadas e consistentes discutida na Subseção 3.2.4. Uma calibração em quatro etapas foi aplicada, sendo as duas primeiras não acuradas e as duas últimas finas, de acordo com a Tabela 4.13. Na primeira etapa, os parâmetros do algoritmo da Figura 3.12 foram mantidos nos valores preliminares **fracaoLimite** = 95%, **nivelMax** = 2 e **maxRk** = 3 que apresentaram resultados aceitáveis embora não estivessem ainda completamente calibrados. O parâmetro γ_2 do algoritmo **FixaVars** da Figura 3.9 foi fixado em 50 para evitar longas reotimizações do algoritmo dual simplex. Os demais parâmetros foram testados para **maxAnalisar** $\in \{2\%; 2{,}5\%; 3\%; 3{,}5\%; 4\%\}$ e **maxFix** $\in \{0{,}5\%; 1\%; 1{,}5\%; 2\%\}$. Os três melhores resultados foram guardados para serem usados na etapa 3 e o melhor resultado foi usado para calibrar a etapa 2. Na etapa 2 foram calibrados os parâmetros **maxRk** e **nivelMax**, mantendo ainda **fracaoLimite** = 95%. Foram testados os valores **maxRk** $\in \{2, 3, 4\}$ e **nivelMax** $\in \{2, 3, 4\}$. A etapa 3 envolveu combinar os três melhores pares de valores da etapa 1 com os três melhores pares de valores da etapa 2. A melhor configuração de parâmetros encontrada foi **maxFix** = 1,5%, **maxAnalisar** = 4%, **nivelMax** = 3 e **maxRk** = 3. Na quarta etapa foram testados valores **fracaoLimite** $\in \{50\%, 70\%, 80\%, 90\%, 95\%, 99\%\}$ e o melhor resultado foi **fracaoLimite** = 95%.

4.3.2 Resultados computacionais de longo prazo

As estratégias de longo prazo podem ser combinadas afetando significativamente a evolução da busca. A Tabela 4.14 estabelece os acrônimos para as estratégias de longo prazo e na Tabela 4.15 estão listadas as 14 diferentes combinações de estratégias testadas. Essa tabela está dividida em três partes. As linhas 1 e 2 consideram o uso da matriz de freqüência com intensificação e diversificação ou apenas com diversificação. As linhas de 3 a 10 tratam as possibilidades de geração de centros de intensificação e diversificação com os algoritmos RK_1 e RK_2. Todas as combinações de geração de centros são aplicadas ao algoritmo RK_1. A execução da linha 3 considera apenas o centro de intensificação. As linhas 4, 5 e 6 consideram apenas centros de diversificação. Um centro de diversificação pode ser gerado de três maneiras diferentes. Pode ser usada a complementação da Subseção 3.2.3, a distância máxima da expressão (3.17) ou a combinação de ambas. A linha 7 considera a melhor estratégia de centro de diversificação com a geração de centro de intensificação. A linha 8 considera a melhor combinação de centros de intensificação e diversificação com o melhor cenário de uso da matriz de freqüência. As linhas 9 e 10 são relativas ao uso do algoritmo RK_2 e apenas as combinações que apresentam os melhores resultados para RK_1 são aplicadas. A linha 9 é relativa à melhor combinação de critérios de geração de centros de intensificação e diversificação e a linha 10 é relativa à combinação de geração de centros com o uso da matriz de freqüência. O terceiro grupo de execuções compreende as linhas de 11 a 14 e é relativo à exploração de variáveis consistentes e fortemente determinadas. As execuções das linhas 11 e 12 usam o algoritmo RK_1 para gerar os subconjuntos $R(k)$ com a expressão (3.16). A execução da linha 11 identifica as variáveis consistentes e fortemente determinadas e a execução da linha 12 incorpora a melhor maneira de usar a matriz de freqüência. As linhas 13 e 14 repetem essas duas execuções para o algoritmo RK_2.

Os resultados computacionais são apresentados nas Tabelas 4.16 e 4.17. A implementação de referência BTP-ACVO é listada nessas tabelas para permitir avaliar o ganho do uso das estratégias de longo prazo. Das 14 execuções listadas na Tabela 4.15, de cada um dos três grupos, a execução que alcançou melhor resultado é apresentada e são: MF_DIV, RK_2–CI e RK_1–SDCV. Nas execuções em que a matriz de freqüência foi utilizada, as instâncias fast0507, nw04 e stp3d não puderam ser resolvidas por esgotamento da memória na tentativa de construir a matriz de freqüência. Em virtude disso, essas instâncias foram desconsideradas quando o teste de Wilcoxon foi aplicado em cenários envolvendo o uso da matriz de freqüência.

O teste de Wilcoxon foi inconclusivo na comparação entre BTP-ACVO e MF_DIV. A BTP-ACVO foi melhor em 14 casos, a MF_DIV foi melhor em 13 e houve 48 empates, totalizando as 75 instâncias consideradas. Se o desvio médio for considerado, MF_DIV consegue um resultado positivo, a execução BTP-ACVO apresenta desvio de 93% e a MF_DIV apresenta 79%. Quando Média(65) é considerada, a BTP-ACVO e a MF_DIV apresentam respectivamente os desvios de 96% e 79%. Na

Tab. 4.14: Acrônimos das estratégias de longo prazo.

Acrônimo	Abordagem
RK_1	Aplica a técnica 1 de geração de centros $R(k)$.
RK_2	Aplica a técnica 2 de geração de centros $R(k)$.
CI	Gera centro de intensificação com RK_1 ou RK_2.
CD_MIN_CC	Gera centro de diversificação usando distância mínima com complementação com RK_1 ou RK_2.
CD_MAX_CC	Gera centro de diversificação usando distância máxima com complementação com RK_1 ou RK_2.
CD_MAX_SC	Gera centro de diversificação usando distância máxima sem complementação com RK_1 ou RK_2.
MF_DIV_INT	Usa matriz de freqüência com intensificação e diversificação.
MF_DIV	Usa matriz de freqüência apenas com diversificação.
SDCV	Explora variáveis consistentes e fortemente determinadas.
MLR{}	O melhor resultado de um conjunto de abordagens.

Tab. 4.15: Execuções de longo prazo efetuadas.

Número	Execução
1	MF_DIV_INT
2	MF_DIV
3	RK_1–CI
4	RK_1–CD_MIN_CC
5	RK_1–CD_MAX_SC
6	RK_1–CD_MAX_CC
7	RK_1–CI–MLR{4, 5, 6}
8	RK_1–MLR{3, 4, 5, 6, 7}–MLR{1, 2}
9	RK_2–MLR{3, 4, 5, 6, 7}
10	RK_2–MLR{3, 4, 5, 6, 7}–MLR{1, 2}
11	RK_1–SDCV
12	RK_1–SDCV–MLR{1, 2}
13	RK_2–SDCV
14	RK_2–SDCV–MLR{1, 2}

comparação da BTP-ACVO com RK_2–CI, o teste de Wilcoxon também se mostrou inconclusivo. A BTP-ACVO vence em 15 casos, a RK_2–CI vence em 12 e há 51 empates. O desvio médio foi melhor para RK_2–CI com 55% enquanto a BTP-ACVO apresenta desvio de 93%. Para Média(65), os desvios são respectivamente 96% e 56% para BTP-ACVO e RK_2–CI. No confronto entre BTP-ACVO e RK_1–SDCV, vence no teste de Wilcoxon a execução RK_1–SDCV. A BTP-ACVO é melhor em 2 instâncias, a RK_1–SDCV é melhor em 27 instâncias e ocorrem 49 empates. O desvio médio para RK_1–SDCV foi de 74% e para Média(65) foi de 76%, sendo também superior à BTP-ACVO quando esse critério é considerado. A execução RK_1–SDCV foi superior tanto à MF_DIV quanto à RK_2–CI com relação ao teste de Wilcoxon. As execuções BTP-ACVO, MF_DIV, RK_2–CI e RK_1–SDCV alcançaram soluções ótimas em respectivamente 22 (28,2%), 21 (26,9%), 20 (25,6%) e 27 (34,6%) instâncias. Da Tabela 4.17, observa-se que para desvio não superior a 5%, a BTP-ACVO atinge esse resultado para 60,3% das instâncias e as execuções de longo prazo MF_DIV, RK_2–CI e RK_1–SDCV atingem respectivamente 53,8%, 55,1% e 69,3%, mostrando novamente a superioridade da execução RK_1–SDCV que inclusive apresentou desvio não superior a 1% em 52,6% das instâncias.

Tab. 4.16: Resultados de longo prazo.

Instâncias		BTP-ACVO		MF_DIV		RK_2–CI		RK_1–SDCV	
nomes	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
10teams	924	**924**	**0**	**924**	**0**	**924**	**0**	**924**	**0**
a1c1s1	11503,4	16757,2	46	16757,2	46	16757,2	46	16757,2	46
aflow30a	1158	1227	6	1294	12	1259	9	1165	1
aflow40b	1168	2358	102	2171	86	2157	85	1927	65
air04	56137	56138	0^+	56138	0^+	56138	0^+	56138	0^+
air05	26374	26444	0^+	26444	0^+	26444	0^+	26444	0^+
cap6000	-2,45E+06	-2,45E+06	0^+	-2,45E+06	0^+	-2,45E+06	0^+	-2,45E+06	0^+
dano3mip	714,125	743,9	4	743,9	4	743,9	4	743,9	4
danoint	65,67	67	2	70,5	7	70	7	**65,67**	**0**
disctom	-5000	–	–	–	–	–	–	–	–
ds	412,503	692,795	68	692,795	68	692,795	68	692,795	68
fast0507	174	175	1	–	–	175	1	175	1
fiber	405935	423613	4	430875	6	446758	10	407942	0^+
fixnet6	3983	3985	0^+	**3983**	**0**	4018	1	**3983**	**0**
glass4	1,20E+09	2,70E+09	125	2,70E+09	125	2,70E+09	125	2,70E+09	125
harp2	-7,39E+07	-7,31E+07	1	-7,29E+07	1	-7,28E+07	1	-7,30E+07	1
liu	1496	2096	40	2096	40	2096	40	2096	40
markshare1	1	18	1700	19	1800	9	800	16	1500
markshare2	1	34	3300	23	2200	18	1700	25	2400
mas74	11801,2	13039,8	10	13230,8	12	12843,9	9	12039,5	2
mas76	40005,1	40121,9	0^+	40501,9	1	40270,3	1	**40005,1**	**0**
misc07	2810	**2810**	**0**	**2810**	**0**	**2810**	**0**	**2810**	**0**
mkc	-563,85	-437,49	22	-437,49	22	-437,49	22	-437,49	22
mod011	-5,46E+07	-5,31E+07	3	-5,37E+07	2	-5,40E+07	1	-5,44E+07	0^+
modglob	2,07E+07	2,08E+07	0^+	2,08E+07	0^+	2,09E+07	1	2,08E+07	0^+
momentum1	109143	–	–	–	–	–	–	–	–
net12	214	337	57	337	57	337	57	337	57
nsrand-ipx	51200	56640	11	56640	11	56640	11	56640	11
nw04	16862	**16862**	**0**	–	–	**16862**	**0**	**16862**	**0**
opt1217	-16	**-16**	**0**	**-16**	**0**	**-16**	**0**	**-16**	**0**
p2756	3124	3226	3	3226	3	3226	3	3226	3
pk1	11	**11**	**0**	**11**	**0**	**12**	9	**11**	**0**
pp08a	7350	7810	6	7880	7	7750	5	7480	2
pp08aCUTS	7350	7580	3	7570	3	7510	2	7360	0^+
protfold	-31	-20	35	-20	35	-20	35	-20	35
qiu	-132,873	-36,464	73	-119,654	10	-119,654	10	**-132,873**	**0**
rd-rplusc-21	165395,3	–	–	–	–	–	–	–	–

Continua na próxima página.

Tab. 4.16 – Continuação da página anterior.

Instâncias		BTP-ACVO		MF_DIV		*RK*_2–CI		*RK*_1–SDCV	
nomes	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
set1ch	54537,8	60472,5	11	59231,8	9	59510,8	9	57788	6
seymour	423	428	1	428	1	428	1	428	1
sp97ar	6,75E+08	7,08E+08	5	7,08E+08	5	7,08E+08	5	7,08E+08	5
stp3d	–	–	–	–	–	–	–	–	–
swath	467,407	522,37	12	553,06	18	522,37	12	522,37	12
t1717	195779	286071	46	308548	58	286071	46	286071	46
tr12-30	130596	166120	27	166120	27	166120	27	157801	21
vpm2	13,75	14,25	4	14,5	5	14,25	4	**13,75**	**0**
l152lav	4722	**4722**	**0**	**4722**	**0**	**4722**	**0**	**4722**	**0**
stein45	30	**30**	**0**	**30**	**0**	**30**	**0**	**30**	**0**
ran8x32	5247	5383	3	5416	3	5390	3	5382	3
ran10x26	4270	4373	2	4421	4	4353	2	4425	4
ran12x21	3664	3868	6	3862	5	3954	8	3791	3
ran13x13	3252	**3252**	**0**	3343	3	3363	3	**3252**	**0**
binkar10_1	6742,20	7202,81	7	7118,01	6	7195,95	7	6784,23	1
irp	12159,5	**12159,5**	**0**	**12159,5**	**0**	**12159,5**	**0**	**12159,5**	**0**
mas284	91405,7	**91405,7**	**0**	**91405,7**	**0**	**91405,7**	**0**	**91405,7**	**0**
prod1	-56	-51	9	-51	9	-53	5	-55	2
bc1	3,33836	3,39703	2	3,39703	2	3,37759	1	3,36804	1
bienst1	46,75	**46,75**	**0**	**46,75**	**0**	**46,75**	**0**	**46,75**	**0**
bienst2	54,6	56,5	3	56,5	3	57,6667	6	56,2500	3
dano3_3	576,345	**576,345**	**0**	**576,345**	**0**	**576,345**	**0**	**576,345**	**0**
dano3_4	576,435	**576,435**	**0**	**576,435**	**0**	**576,435**	**0**	**576,435**	**0**
dano3_5	576,925	577,052	0^+	577,052	0^+	577,052	0^+	577,052	0^+
mkc1	-607,15	-604,91	0^+	-605,56	0^+	-604,46	0^+	-605,41	0^+
neos1	19	**19**	**0**	**19**	**0**	**19**	**0**	**19**	**0**
neos2	454,86	1390,65	206	1212,31	167	1301,07	186	1353,98	198
neos3	368,84	1353,98	267	1353,98	267	1353,98	267	1284,56	248
neos4	-4,86E+10	-4,61E+10	5	-4,61E+10	5	-4,61E+10	5	-4,61E+10	5
neos5	15	**15**	**0**	**15**	**0**	**15**	**0**	**15**	**0**
neos6	83	87	5	87	5	87	5	87	5
seymour1	410,764	411,536	0^+	411,536	0^+	411,536	0^+	411,536	0^+
swath1	379,071	**379,071**	**0**	**379,071**	**0**	**379,071**	**0**	**379,071**	**0**
swath2	385,2	**385,2**	**0**	**385,2**	**0**	**385,2**	**0**	**385,2**	**0**
acc-0	0	**0**	–	**0**	–	**0**	–	**0**	–
acc-1	0	**0**	–	**0**	–	**0**	–	**0**	–
acc-2	0	**0**	–	**0**	–	**0**	–	**0**	–
acc-3	0	**0**	–	**0**	–	**0**	–	**0**	–
acc-4	0	–	–	–	–	–	–	–	–
acc-5	0	1	–	1	–	1	–	1	–
acc-6	0	**0**	–	**0**	–	**0**	–	**0**	–
Média			93		79		55		74
Média(65)			96		79		56		76
Falhas			5		7		5		5

4.3.3 Busca tabu paramétrica e heurísticas de busca local

Este capítulo é encerrado ao avaliar a busca tabu paramétrica em relação às heurísticas de busca local. Os resultados reportados na Tabela 4.18 são relativos às execuções do *branch-and-cut* do Cplex, heurística *local branching* (LB) (Fischetti e Lodi, 2003), *relaxation induced neighborhood search* (RINS) (Danna et al., 2005), *distance induced neighborhood search* (DINS) (Ghosh, 2007), BTP-ACVO e *RK*_1–SDCV. Os resultados relativos às heurísticas de busca local foram reportados por Ghosh (2007) ao executá-las por uma hora em um PC com processador AMD Athlon 2,4 MHz com 128 MByte de memória em sistema operacional Redhat Linux 9.0. A versão de Cplex usada por ele foi a 9.13. Como o autor apresenta apenas os desvios em relação aos seus limitantes superiores, esses limitantes foram mantidos para essa comparação e não foi possível apresentar os valores de

Tab. 4.17: Resultados de longo prazo (agregados).

	BTP-ACVO		MF_DIV		RK_2–CI		RK_1–SDCV	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	32	41,0	28	35,9	31	39,7	41	52,6
1% a 5%	15	19,2	14	17,9	12	15,4	13	16,7
5% a 10%	6	7,7	10	12,8	12	15,4	2	2,6
10% a 25%	5	6,4	5	6,4	4	5,1	4	5,1
25% a 50%	5	6,4	4	5,1	5	6,4	4	5,1
50% a 100 %	3	3,8	4	5,1	3	3,8	3	3,8
Acima de 100%	6	7,7	5	6,4	5	6,4	5	6,4
Não ótimas em Accs	1	1,3	1	1,3	1	1,3	1	1,3
Falhas	5	6,4	7	9,0	5	6,4	5	6,4

função objetivo para as heurísticas de busca local. Das instâncias utilizadas por Ghosh (2007), foram excluídas aquelas com variáveis inteiras gerais e as instâncias rail2586c, rail4284c, rail4872c, ljb2, ljb7, ljb9, ljb10 e ljb12 que não foram disponibilizadas pelo autor e nem encontradas na Internet. Algumas informações relativas às 39 instâncias utilizadas podem ser encontradas na Tabela C.2 do Apêndice C. Há 19 instâncias comuns com as da Miplib apresentadas anteriormente neste capítulo, mas a coluna "Melhor Valor" pode diferir pois foi preciso manter aqui os limitantes apresentados por Ghosh (2007).

O teste de Wilcoxon é inconclusivo na comparação do Cplex com a LB. O Cplex vence em 12 casos, a LB vence em 20 e há 7 empates. Por esse mesmo critério, a heurística RINS é superior tanto ao Cplex quanto à LB. Na comparação com o Cplex, a RINS vence em 24 casos, perde em 8 e ocorrem 7 empates. Na comparação com a LB, a RINS vence em 23 casos, perde em 10 e ocorrem 6 empates. A heurística DINS foi superior ao Cplex e à LB com relação ao teste de Wilcoxon e o resultado foi inconclusivo quando comparada com a RINS. A DINS foi melhor que o Cplex, LB e RINS em respectivamente 25, 28 e 19 casos. Perdeu em 5, 6 e 13 casos e houve 9, 5, e 7 empates respectivamente. Pelo teste de Wilcoxon, as execuções BTP-ACVO e RK_1–SDCV são dominadas quando comparadas com o Cplex ou com as heurísticas de busca local. A BTP-ACVO foi melhor que o Cplex apenas para as instâncias dc1c e swath. A RK_1–SDCV empata para a instância danoint e é melhor para as instâncias dc1c e swath. Com relação à LB, a BTP-ACVO foi superior para a instância ds e empata para a instância fast0507. A RK_1–SDCV empata com LB para as instâncias fast0507 e danoint e foi superior para as instâncias ds e dc1l. Quando a BTP-ACVO e a RK_1–SDCV são comparadas com a heurística RINS, a BTP-ACVO consegue apenas um empate para a instância fast0507 e a RK_1–SDCV consegue empates para as instâncias fast0507 e danoint. A execução BTP-ACVO não consegue nenhum resultado positivo em relação à DINS e a execução RK_1–SDCV

consegue apenas um empate para a instância danoint. A execução RK_1–SDCV supera a BTP-ACVO quando o teste de Wilcoxon é considerado. A execução RK_1–SDCV ganha em 15 instâncias, perde em 7 e ocorrem 17 empates. A execução RK_1-SDCV consegue baixar o desvio médio de 617,361% para 562,944%. As demais execuções Cplex, LB, RINS e DINS apresentaram respectivamente os desvios médios 66,786%, 69,909%, 63,018% e 60,199%. Quando o desvio não superior a 5% é considerado, a Tabela 4.19 revela que Cplex, LB, RINS, DINS, BTP-ACVO e RK_1–SDCV atingem respectivamente 74,4%, 74,4%, 84,6%, 87,2%, 17,9% e 17,9%. Com esse critério de julgamento, a heurística DINS alcança um valor expressivo seguido de perto pela RINS. O resultado negativo da BTP em relação ao Cplex ou heurísticas de busca local embutidas no algoritmo de *branch-and-cut* era esperado e algumas propostas de integração da BTP com *branch-and-cut* são apresentadas na conclusão.

Tab. 4.18: Heurísticas de busca local × BTP.

Instâncias		Cplex	LB	RINS	DINS	BTP-ACVO		RK_1–SDCV	
nomes	Melhor Valor	Desv	Desv	Desv	Desv	Objetivo	Desv	Objetivo	Desv
a1c1s1	11505,43	2,347	0,250	0,000	0,079	16757,18	45,646	16757,18	45,646
a2c1s1	10889,13	2,978	1,889	0,000	0,024	17002,61	56,143	16036,80	47,274
b1c1s1	24544,25	5,977	1,786	0,933	4,444	49221,31	100,541	44466,31	81,168
b2c1s1	25740,15	4,240	2,701	0,559	1,010	49847,39	93,656	50189,61	94,986
biella1	3,07E+06	0,309	0,806	0,426	0,739	3,10E+06	1,127	3,15E+06	2,912
danoint	65,67	0,000	0,000	0,000	0,000	67	2,025	65,67	0,005
glass4	1,46E+09	13,014	7,534	2,740	4,794	2,70E+09	84,931	2,70E+09	84,931
markshare1	1	500,000	400,000	400,000	500,000	18	1700,000	16	1500,000
markshare2	1	1300,000	1100,000	2000,000	1800,000	34	3300,000	25	2400,000
mkc	-563,846	0,180	0,049	0,043	0,021	-437,49	22,410	-437,49	22,410
net12	214	0,000	0,000	0,000	0,000	337	57,477	337	57,477
nsrand-ipx	51200	0,625	0,625	0,313	0,000	56640	10,625	56640	10,625
rail507	174	0,000	0,000	0,000	0,000	177	1,724	177	1,724
seymour	423	0,473	0,473	0,000	0,236	428	1,182	428	1,182
sp97ar	6,62E+08	0,428	0,513	0,335	0,000	7,08E+08	7,058	7,08E+08	7,058
sp97ic	4,28E+08	0,793	0,642	0,551	0,000	4,74E+08	10,806	4,68E+08	9,236
sp98ar	5,30E+08	0,184	0,106	0,177	0,228	5,68E+08	7,226	5,58E+08	5,237
sp98ic	4,49E+08	0,270	0,146	0,204	0,072	4,86E+08	8,132	4,82E+08	7,312
swath	471,03	18,067	5,679	8,089	4,622	522,3699	10,899	522,3699	10,899
tr12-30	130596	0,000	0,024	0,000	0,000	166120	27,201	157801	20,831
berlin-5-8-0	62	0,000	0,000	0,000	0,000	68	9,677	69	11,290
bg512142	189183,2	7,257	5,192	0,161	0,000	2,55E+07	13398,802	2,55E+07	13398,802
blp-ic97	4048,35	0,779	0,653	0,358	0,000	4601,59	13,666	4423,71	9,272
blp-ic98	4494,68	0,961	1,056	0,746	0,515	5480,15	21,925	5392,49	19,975
blp-ar98	6211,45	0,655	0,060	0,461	0,000	6945,513	11,818	6997,664	12,657
cms750-4	253	2,372	0,791	1,186	0,791	335	32,411	335	32,411
dc1c	1,84E+06	695,213	2,353	0,296	0,773	2,11E+06	14,330	2,11E+06	14,330
dc1l	1,81E+06	2,018	8,166	6,994	1,572	1,96E+06	8,222	1,96E+06	8,068
dg012142	2,71E+06	17,457	25,984	4,963	3,943	1,24E+08	4494,827	9,85E+07	3539,635
railway-8-1-0	400	0,250	0,000	0,250	0,250	415	3,750	412	3,000
trento1	5,19E+06	0,000	193,118	1,912	0,402	1,58E+07	204,104	1,63E+07	214,914
usabbrv-8-25-70	121	3,306	2,479	0,000	1,653	158	30,579	160	32,231
aflow40b	1168	0,257	1,455	0,000	0,000	2358	101,884	1927	64,983
dano3mip	688,26	2,602	3,595	4,724	2,230	743,9	8,084	743,9	8,084
ds	413,78	11,226	945,745	11,226	6,119	692,795	67,431	692,795	67,431
fast0507	174	0,000	0,575	0,575	0,000	175	0,575	175	0,575
harp2	-7,39E+07	0,001	0,001	0,023	0,000	-7,31E+07	1,133	-7,30E+07	1,222
liu	1212	2,475	10,066	3,465	5,281	2096	72,937	2096	72,937
t1717	216557	7,948	1,939	5,979	7,948	286071	32,100	286071	32,100
Média		66,786	69,909	63,018	60,199		617,361		562,944

Tab. 4.19: Heurísticas de busca local × BTP (agregados).

	Cplex		LB		RINS		DINS		BTP-ACVO		*RK*_1–SDCV	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	21	53,8	20	51,3	27	69,2	26	66,7	1	2,6	2	5,1
1% a 5%	8	20,5	9	23,1	6	15,4	8	20,5	6	15,4	5	12,8
5% a 10%	3	7,7	4	10,3	3	7,7	3	7,7	6	15,4	7	17,9
10% a 25%	4	10,3	1	2,6	1	2,6	0	0,0	8	20,5	8	20,5
25% a 50%	0	0,0	1	2,6	0	0,0	0	0,0	5	12,8	5	12,8
50% a 100 %	0	0,0	0	0,0	0	0,0	0	0,0	6	15,4	7	17,9
Acima de 100%	3	7,7	4	10,3	2	5,1	2	5,1	7	17,9	5	12,8

Capítulo 5

Resultados Computacionais Adicionais

A avaliação do desempenho da busca tabu paramétrica nas instâncias da Miplib apresentada no Capítulo 4 mostra a robustez da implementação em instâncias de problemas de diversas áreas. Neste capítulo, a ênfase concentra-se em avaliar o desempenho da heurística para instâncias de um mesmo problema. Para essa finalidade, foram consideradas 32 instâncias do problema da mochila multidimensional (*multidimensional knapsack problem*, MKP), 32 instâncias do problema da mochila multidimensional com restrições de demanda (*multidemand multidimensional knapsack problem*, MDMKP) e 57 instâncias do problema de designação generalizada, (*generalized assignment problem*, GAP).

A forma de apresentação dos resultados deste capítulo segue o mesmo padrão utilizado no Capítulo 4. A mesma calibração de parâmetros usada no Capítulo 4 foi mantida para os três conjuntos de instâncias. Primeiramente, são apresentados os modelos matemáticos e as instâncias utilizadas, e na seqüência os resultados computacionais. O Apêndice D mostra algumas informações adicionais relativas aos testes de Wilcoxon que foram realizados.

5.1 Modelos matemáticos e instâncias utilizadas

Esta seção descreve brevemente os modelos matemáticos dos três problemas considerados e a origem das instâncias utilizadas para avaliar a busca tabu paramétrica.

5.1.1 Problema da mochila multidimensional

O problema da mochila multidimensional envolve encontrar um subconjunto de um total de n itens que maximize a soma das utilidades enquanto satisfaz m restrições de capacidade. O modelo

matemático de programação inteira é o descrito a seguir:

$$\max \sum_{j=1}^{n} c_j x_j \tag{5.1}$$

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, ..., m \tag{5.2}$$

$$x_j \in \{0, 1\}, \quad j = 1, ..., n. \tag{5.3}$$

Na função objetivo (5.1), c_j representa a utilidade de cada item j. As restrições (5.2) representam as m restrições de mochila em que a_{ij} é o volume ocupado pelo item j na mochila i e b_i é a capacidade da mochila i que não pode ser excedida. As restrições (5.3) expressam o domínio das variáveis e representam a exclusão ou inclusão de cada item j nas m mochilas.

Chu e Beasley (1998) geraram uma biblioteca de instâncias com $n \in \{100, 250, 500\}$ e $m \in \{5, 10, 30\}$. Os coeficientes a_{ij} são inteiros gerados aleatoriamente entre 0 e 1000. Os valores b_i são tais que $b_i = \alpha \sum_{j=1}^{n} a_{ij}$ em que α é um parâmetro que controla a capacidade de uma restrição de mochila (5.2). Essa imposição diminui com o crescimento de α. Para cada um dos 9 pares m–n, foram geradas 10 instâncias para cada valor de $\alpha \in \{0{,}25; 0{,}50; 0{,}75\}$, resultando em 30 instâncias para cada par m–n e totalizando 270 instâncias na biblioteca.

As instâncias escolhidas para serem tratadas pela busca tabu paramétrica foram selecionadas dos grupos que Chu e Beasley (1998) relatam como sendo os mais difíceis de serem resolvidos. Para cada par m–n com $m \in \{5, 10, 30\}$ e $n \in \{100, 500\}$ foram escolhidas 4 instâncias considerando $\alpha = 0{,}25$ para $n = 100$ e $\alpha = 0{,}50$ para $n = 500$. Também foram selecionadas 4 instâncias do grupo 30–500 com $\alpha = 0{,}75$ e 4 instâncias do grupo 5–250 com $\alpha = 0{,}25$, totalizando 32 instâncias. As instâncias são identificadas com a nomenclatura *mkp–m–n–s*, em que *s* é o número da instância dentro do grupo m–n.

5.1.2 Problema da mochila multidimensional com restrições de demanda

Quando no problema da mochila multidimensional são adicionadas restrições de demanda, surge o problema da mochila multidimensional com restrições de demanda. O seu modelo matemático de programação inteira é descrito por:

$$\max \sum_{j=1}^{n} c_j x_j \tag{5.4}$$

$$\sum_{j=1}^{n} a_{ij}x_j \leq b_i, \quad i = 1, ..., m \tag{5.5}$$

$$\sum_{j=1}^{n} a_{ij}x_j \geq b_i, \quad i = m+1, ..., m+q \tag{5.6}$$

$$x_j \in \{0, 1\}, \quad j = 1, ..., n. \tag{5.7}$$

A função objetivo, as restrições de mochila e as restrições de integralidade das variáveis de decisão das expressões (5.4), (5.5) e (5.7) do MDMKP são equivalentes às expressões (5.1), (5.2) e (5.3) do MKP. A diferença são as q restrições de demanda da expressão (5.6) que são do tipo "maior ou igual que". Encontrar uma solução inteira-factível é uma tarefa desafiadora para várias instâncias desse problema pois as restrições de demanda são conflitantes com as restrições de mochila e reduzem significativamente o espaço de soluções em relação ao MKP.

Cappanera e Trubian (2005) geraram uma biblioteca de instâncias de MDMKP modificando as instâncias de MKP criadas por Chu e Beasley (1998). Foram escolhidas 5 instâncias de cada configuração m–n–α e a partir de cada uma foram geradas 6 instâncias, usando a função objetivo com todos os coeficientes positivos ou com coeficientes positivos e negativos. Foram utilizados valores de q tais que $q \in \{1, m/2, m\}$.

As instâncias escolhidas para serem tratadas pela busca tabu paramétrica foram selecionadas dos grupos derivados de mkp–5–100, mkp–5–500, mkp–30–100 e mkp–30–500. Para cada valor de $q \in \{m/2, m\}$ foram selecionadas 4 instâncias, sendo duas com todos os coeficientes positivos na função objetivo e duas com coeficientes positivos e negativos, totalizando 32 instâncias. A nomenclatura usada para designar as instâncias é a mesma de Arntzen et al., (2006). Assim, em *mdmkp–m–n–q–r–s*, *r* é um se houver coeficientes negativos na instância e *s* é o número da instância dentro de um grupo.

5.1.3 Problema de designação generalizada

O problema de designação generalizada envolve encontrar a designação de cada uma de n tarefas a exatamente um dentre m agentes com custo mínimo. O modelo matemático de programação inteira é descrito por:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}x_{ij} \tag{5.8}$$

$$\sum_{i=1}^{m} x_{ij} = 1, \quad j = 1, ..., n \tag{5.9}$$

$$\sum_{j=1}^{n} a_{ij} x_{ij} \leq b_i, \quad i = 1, ..., m \tag{5.10}$$

$$x_{ij} \in \{0, 1\}, \quad i = 1, ..., m, j = 1, ..., n. \tag{5.11}$$

Na função objetivo dada pela expressão (5.8), c_{ij} é o custo de designar a tarefa j ao agente i. As restrições (5.9) em conjunto com as restrições de integralidade (5.11) garantem que cada tarefa j seja designada a exatamente um agente i. As restrições (5.10) garantem que a capacidade máxima b_i de cada agente não seja violada, dado que designar uma tarefa j a um agente i consome a quantidade de recurso a_{ij}.

Chu e Beasley (1997) geraram uma biblioteca de instâncias com $m \in \{5, 10, 15, 20, 40, 60, 80\}$ e $n \in \{100, 200, 400, 900, 1600\}$. O autores usaram diferentes critérios de geração de instâncias e as agruparam em quatro conjuntos A, B, C e D. Yagiura et al. (2006) estenderam a biblioteca incluindo o conjunto E. As instâncias do grupo A são consideradas as mais fáceis, seguidas das do grupo B. As instâncias dos grupos D e E são consideradas as mais difíceis pois os custos de designação e os recursos consumidos são inversamente correlacionados. Todas as 57 instâncias distribuídas entre os grupos de A a E foram utilizadas para validar o desempenho da busta tabu paramétrica. A nomenclatura usada para designar as instâncias é a seguinte: o primeiro caractere indica o grupo da instância, os próximos dois caracteres indicam o número de agentes e os restantes indicam o número de tarefas. Por exemplo, E801600 trata-se de uma instância do grupo E, com 80 agentes e 1600 tarefas.

5.2 Resultados com uso de memória de curto prazo

Nesta seção, para os três problemas considerados, são apresentados os resultados para as estratégias de cálculo de penalidade inteira e resistência a metas e na seqüência os resultados de curto prazo. Os dados da coluna "Melhor Valor" para as instâncias de GAP foram selecionados de Yagiura et al. (2006) e da execução do *branch-and-cut* do Cplex 10 por duas horas, sempre reportando o melhor dos dois valores. Para as instâncias de MKP, os melhores valores de função objetivo conhecidos vêm de Chu e Beasley (1998) e da execução do Cplex por duas horas. Para as instâncias de MDMKP, a execução do Cplex por duas horas foi aplicada para encontrar o melhor valor. Para essas instâncias, houve 10 casos em que a BTP apresentou resultados melhores que os do *branch-and-cut* do Cplex. Quando isso ocorre, o valor encontrado pela BTP é o usado e o nome da instância aparece em negrito.

Os mesmos critérios de avaliação usados no Capítulo 4 também são aqui aplicados considerando execuções com limite de tempo de 3600s. Em primeiro lugar é aplicado o teste de classificação com sinal de Wilcoxon. Em segundo lugar a execução que falha em menos instâncias. E em terceiro lugar, o número de instâncias cujos desvios ficam abaixo de 5%.

5.2.1 Tratamento da penalidade inteira e da resistência a metas

Nesta subseção são avaliadas as técnicas de cálculo de penalidade inteira e resistência a metas, ACVO, DM e RB. As Tabelas 5.1 e 5.2 trazem os resultados das instâncias de GAP, as Tabelas 5.3 e 5.4 o resultados das instâncias de MKP e as Tabelas 5.5 e 5.6 os resultados das instâncias de MDMKP. Para o problema GAP, o uso da ACVO foi superior ao uso da DM e da RB pelo teste de Wilcoxon. A ACVO falha em 3 instâncias, a DM em 6 e a RB em 7. Entre DM e RB, o teste de Wilcoxon é inconclusivo. As três execuções encontram a solução ótima para 8 instâncias. O desvio médio para ACVO é um pouco superior inclusive quando Média(50) é considerada. Para desvio não superior a 5%, a ACVO também é vantajosa com 78,9% das instâncias dentro dessa faixa, enquanto DM e RB apresentam respectivamente 68,4% e 71,9%.

Para o problema MKP, o teste de Wilcoxon foi inconclusivo entre todas as execuções. Em instâncias de MKP, a BTP sempre foi capaz de encontrar pelo menos uma solução inteira-factível, independente da execução realizada. A ACVO encontra 6 soluções ótimas, a DM encontra 4 e a RB encontra 5. Os desvios médios são muito próximos entre si com uma pequena vantagem para a execução RB. Como todas as instâncias apresentam desvio inferior a 5% para MKP, para esse problema a discussão será modificada para desvio não superior a 1%. A ACVO alcança essa marca para 84,4% das instâncias, a DM para 87,5% e a RB para 90,6%.

Para o problema MDMKP, o teste de Wilcoxon também foi inconclusivo entre todas as execuções. As três execuções encontram 3 soluções ótimas. A ACVO falha em 5 instâncias, a DM em 4 e a RB em 4. O desvio médio para a execução ACVO apresentou melhor resultado e quando Média(27) é usada a ACVO ainda mantém uma pequena vantagem. Para desvio não superior a 5%, a ACVO atinge essa faixa para 65,6% das instâncias e a DM e RB atingem essa faixa para respectivamente 71,9% e 62,5% das instâncias.

Assim como ocorre para as instâncias da Miplib, a técnica ACVO é a mais indicada para o GAP. Para MKP e MDMKP, as melhores técnicas são respectivamente a RB e a DM. A ACVO embora não tenha sido sempre a vencedora, ainda assim apresentou resultados competitivos. Nas execuções de curto prazo, extensões de curto prazo e longo prazo, foi mantida a implementação de referência com uso da abordagem ACVO.

Tab. 5.1: GAP – Cálculo da penalidade inteira e da resistência a metas.

Instâncias		ACVO		DM		RB	
Nomes	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
A05100	1698	**1698**	**0,0**	**1698**	**0,0**	**1698**	**0,0**
A05200	3235	**3235**	**0,0**	**3235**	**0,0**	**3235**	**0,0**
A10100	1360	**1360**	**0,0**	**1360**	**0,0**	**1360**	**0,0**
A10200	2623	**2623**	**0,0**	**2623**	**0,0**	**2623**	**0,0**
A20100	1158	**1158**	**0,0**	**1158**	**0,0**	**1158**	**0,0**
A20200	2339	**2339**	**0,0**	**2339**	**0,0**	**2339**	**0,0**
B05100	1843	**1843**	**0,0**	**1843**	**0,0**	1845	0,1
B05200	3552	3591	1,1	3590	1,1	3593	1,2
B10100	1407	1412	0,4	1412	0,4	**1407**	**0,0**
B10200	2827	2870	1,5	2898	2,5	2888	2,2
B20100	1166	1187	1,8	1188	1,9	1189	2,0
B20200	2339	2356	0,7	2351	0,5	2359	0,9
C05100	1931	**1931**	**0,0**	**1931**	**0,0**	**1931**	**0,0**
C05200	3456	3493	1,1	3484	0,8	3488	0,9
C10100	1402	1482	5,7	1485	5,9	1469	4,8
C10200	2806	2870	2,3	2874	2,4	2866	2,1
C10400	5597	5660	1,1	5670	1,3	5667	1,3
C15900	11340	11435	0,8	11469	1,1	11485	1,3
C20100	1243	1298	4,4	1334	7,3	1302	4,7
C201600	18802	18975	0,9	18905	0,5	19014	1,1
C20200	2391	2443	2,2	2463	3,0	2446	2,3
C20400	4782	4853	1,5	4867	1,8	4851	1,4
C30900	9982	10095	1,1	10100	1,2	10127	1,5
C401600	17145	17283	0,8	17320	1,0	17658	3,0
C40400	4244	4288	1,0	4290	1,1	4293	1,2
C60900	9327	9409	0,9	9435	1,2	9971	6,9
C801600	16285	16410	0,8	–	–	–	–
D05100	6353	6533	2,8	6535	2,9	6554	3,2
D05200	12742	12885	1,1	13029	2,3	12889	1,2
D10100	6348	6753	6,4	6722	5,9	6754	6,4
D10200	12432	12835	3,2	12663	1,9	12844	3,3
D10400	24693	25238	2,2	25357	2,7	25293	2,4
D15900	55409	55848	0,8	55917	0,9	55991	1,1
D20100	6190	6625	7,0	6947	12,2	6603	6,7
D201600	97832	98279	0,5	98752	0,9	98329	0,5
D20200	12241	12877	5,2	13100	7,0	12905	5,4
D20400	24574	25149	2,3	25421	3,4	25176	2,4
D30900	54852	55375	1,0	56713	3,4	55288	0,8
D401600	97105	98004	0,9	99959	2,9	–	–
D40400	24392	25262	3,6	26474	8,5	25321	3,8
D60900	54568	55802	2,3	–	–	–	–
D801600	97035	–	–	–	–	–	–
E05100	12681	12721	0,3	12708	0,2	12756	0,6
E05200	24930	24977	0,2	24942	0^+	25285	1,4
E10100	11577	12069	4,2	12209	5,5	12725	9,9
E10200	23307	24897	6,8	24090	3,4	24048	3,2
E10400	45746	47112	3,0	47763	4,4	47815	4,5
E15900	102421	103645	1,2	105563	3,1	105287	2,8
E20100	8436	9630	14,2	10449	23,9	10749	27,4
E201600	180646	182775	1,2	184980	2,4	185565	2,7
E20200	22379	24597	9,9	26876	20,1	25424	13,6
E20400	44877	47125	5,0	50693	13,0	48506	8,1
E30900	100429	103442	3,0	106989	6,5	104583	4,1
E401600	178294	182929	2,6	–	–	–	–
E40400	44562	49890	12,0	59273	33,0	52073	16,9
E60900	100153	–	–	–	–	–	–
E801600	176828	–	–	–	–	–	–
Média			2,5		4,0		3,4
Média(50)			2,5		4,0		3,4
Falhas			3		6		7

Tab. 5.2: GAP – Cálculo da penalidade inteira e da resistência a metas (agregados).

	ACVO		DM		RB	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	21	36,8	16	28,1	14	24,6
1% a 5%	24	42,1	23	40,4	27	47,4
5% a 10%	7	12,3	7	12,3	6	10,5
10% a 25%	2	3,5	4	7,0	2	3,5
25% a 50%	0	0,0	1	1,8	1	1,8
50% a 100 %	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0
Falhas	3	5,3	6	10,5	7	12,3

Tab. 5.3: MKP – Cálculo da penalidade inteira e da resistência a metas.

Instâncias		ACVO		DM		RB	
Nomes	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
mkp-5-100-00	24381	**24381**	**0,0**	**24381**	**0,0**	**24381**	**0,0**
mkp-5-100-01	24274	**24274**	**0,0**	**24274**	**0,0**	**24274**	**0,0**
mkp-5-100-02	23551	**23551**	**0,0**	**23551**	**0,0**	23538	0,1
mkp-5-100-03	23534	**23534**	**0,0**	**23534**	**0,0**	**23534**	**0,0**
mkp-5-250-00	59312	59215	0,2	59206	0,2	59224	0,1
mkp-5-250-01	61472	61391	0,1	61468	0^+	61468	0^+
mkp-5-250-02	62130	61719	0,7	62009	0,2	**62130**	**0,0**
mkp-5-250-03	59463	57858	2,7	59430	0,1	59437	0^+
mkp-5-500-10	218428	218281	0,1	218313	0,1	218320	0^+
mkp-5-500-11	221202	220895	0,1	220437	0,3	220588	0,3
mkp-5-500-12	217542	216994	0,3	217019	0,2	217453	0^+
mkp-5-500-13	223560	223487	0^+	223326	0,1	223491	0^+
mkp-10-100-00	23064	23025	0,2	23052	0,1	23050	0,1
mkp-10-100-01	22801	22672	0,6	22704	0,4	22753	0,2
mkp-10-100-02	22131	**22131**	**0,0**	22021	0,5	22040	0,4
mkp-10-100-03	22772	**22772**	**0,0**	22580	0,8	**22772**	**0,0**
mkp-10-500-10	217377	216001	0,6	216996	0,2	216428	0,4
mkp-10-500-11	219066	217840	0,6	217807	0,6	217823	0,6
mkp-10-500-12	217797	217130	0,3	217013	0,4	216927	0,4
mkp-10-500-13	216868	215704	0,5	216065	0,4	216078	0,4
mkp-30-100-00	21946	21367	2,6	21361	2,7	21428	2,4
mkp-30-100-01	21716	21322	1,8	21129	2,7	21519	0,9
mkp-30-100-02	20754	20385	1,8	20266	2,4	20222	2,6
mkp-30-100-03	21464	20849	2,9	20780	3,2	20843	2,9
mkp-30-500-10	218068	216669	0,6	217740	0,2	217746	0,1
mkp-30-500-11	214626	213743	0,4	213638	0,5	213415	0,6
mkp-30-500-12	215914	215177	0,3	214727	0,5	215275	0,3
mkp-30-500-13	217863	217292	0,3	216784	0,5	216562	0,6
mkp-30-500-20	301656	300565	0,4	300549	0,4	300675	0,3
mkp-30-500-21	300048	298981	0,4	299378	0,2	299333	0,2
mkp-30-500-22	305038	304865	0,1	303969	0,4	304471	0,2
mkp-30-500-23	302004	301554	0,1	301530	0,2	301066	0,3
Média			0,6		0,6		0,5

Tab. 5.4: MKP – Cálculo da penalidade inteira e da resistência a metas (agregados).

	ACVO		DM		RB	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	27	84,4	28	87,5	29	90,6
1% a 5%	5	15,6	4	12,5	3	9,4
5% a 10%	0	0,0	0	0,0	0	0,0
10% a 25%	0	0,0	0	0,0	0	0,0
25% a 50%	0	0,0	0	0,0	0	0,0
50% a 100 %	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0

Tab. 5.5: MDMKP – Cálculo da penalidade inteira e da resistência a metas.

Instâncias		ACVO		DM		RB	
Nomes	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
mdmkp-05-100-2-0-0	28384	28205	0,6	28228	0,5	28268	0,4
mdmkp-05-100-2-0-13	67147	**67147**	**0,0**	**67147**	**0,0**	**67147**	**0,0**
mdmkp-05-100-2-1-14	19614	19558	0,3	19589	0,1	19589	0,1
mdmkp-05-100-2-1-2	10124	9797	3,2	9651	4,7	9481	6,4
mdmkp-05-100-5-0-1	26280	**26280**	**0,0**	**26280**	**0,0**	**26280**	**0,0**
mdmkp-05-100-5-0-14	53386	**53386**	**0,0**	**53386**	**0,0**	**53386**	**0,0**
mdmkp-05-100-5-1-0	10263	9687	5,6	9754	5,0	9490	7,5
mdmkp-05-100-5-1-14	17346	17084	1,5	16952	2,3	16983	2,1
mdmkp-05-500-2-0-11	364332	362202	0,6	362508	0,5	362257	0,6
mdmkp-05-500-2-0-4	157693	155531	1,4	155776	1,2	155667	1,3
mdmkp-05-500-2-1-14	113353	111914	1,3	112060	1,1	112153	1,1
mdmkp-05-500-2-1-3	55413	54046	2,5	53635	3,2	54484	1,7
mdmkp-05-500-5-0-1	145350	144783	0,4	143972	0,9	144460	0,6
mdmkp-05-500-5-0-4	130928	129640	1,0	127912	2,3	129828	0,8
mdmkp-05-500-5-1-14	99513	97420	2,1	97379	2,1	96990	2,5
mdmkp-05-500-5-1-8	87420	85385	2,3	85223	2,5	85424	2,3
mdmkp-30-100-15-0-11	39106	36643	6,3	37787	3,4	37136	5,0
mdmkp-30-100-15-0-8	28490	26628	6,5	26579	6,7	26736	6,2
mdmkp-30-100-15-1-12	18519	15016	18,9	14864	19,7	14627	21,0
mdmkp-30-100-15-1-4	7118	–	–	2849	60,0	3864	45,7
mdmkp-30-100-30-0-12	31988	–	–	–	–	–	–
mdmkp-30-100-30-0-5	23296	–	–	–	–	–	–
mdmkp-30-100-30-1-12	10751	–	–	–	–	–	–
mdmkp-30-100-30-1-3	3628	–	–	–	–	–	–
mdmkp-30-500-15-0-13	242660	239302	1,4	239094	1,5	239044	1,5
mdmkp-30-500-15-0-5	159321	155773	2,2	155908	2,1	155584	2,3
mdmkp-30-500-15-1-11	102997	99765	3,1	99245	3,6	99610	3,3
mdmkp-30-500-15-1-2	50134	46411	7,4	46131	8,0	45996	8,3
mdmkp-30-500-30-0-13	214221	211801	1,1	211704	1,2	211215	1,4
mdmkp-30-500-30-0-7	156343	152810	2,3	153075	2,1	152704	2,3
mdmkp-30-500-30-1-14	87187	83296	4,5	83178	4,6	83417	4,3
mdmkp-30-500-30-1-2	48781	44893	8,0	44606	8,6	44782	8,2
Média			3,1		5,3		4,9
Média(27)			3,1		3,3		3,4
Falhas			5		4		4

Tab. 5.6: MDMKP – Cálculo da penalidade inteira e da resistência a metas (agregados).

	ACVO		DM		RB	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	8	25,0	7	21,9	8	25,0
1% a 5%	13	40,6	16	50,0	12	37,5
5% a 10%	5	15,6	3	9,4	6	18,8
10% a 25%	1	3,1	1	3,1	1	3,1
25% a 50%	0	0,0	0	0,0	1	3,1
50% a 100 %	0	0,0	1	3,1	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0
Falhas	5	15,6	4	12,5	4	12,5

5.2.2 Resultados computacionais de curto prazo

Nesta subseção é avaliada a busca tabu paramétrica de curto prazo. A execução BTP-ACVO(1^{st}) é relativa à execução que pára tão logo a primeira solução inteira-factível seja encontrada, ou após o tempo limite de 3600s. A execução BTP-ACVO pára apenas após o tempo limite de 3600s. A execução HCPLEX é relativa às heurísticas do Cplex no nó raiz com limite de tempo de 3600s. Como não há resultados da OFP reportados para GAP, MKP e MDMKP, a BTP foi comparada apenas com a HCPLEX. As Tabelas 5.7 e 5.8 trazem os resultados das instâncias de GAP, as Tabelas 5.9 e 5.10 o resultados das instâncias de MKP e as Tabelas 5.11 e 5.12 os resultados das instâncias de MDMKP.

Para o problema GAP, a execução BTP-ACVO(1^{st}) perde no teste de Wilcoxon para a HCPLEX. A HCPLEX é mais rápida em 43 casos, é mais lenta em 5 e ambas gastam o mesmo tempo em 9 casos. O teste de Wilcoxon envolvendo BTP-ACVO e HCPLEX foi inconclusivo e a BTP-ACVO supera a BTP-ACVO(1^{st}) no teste de Wilcoxon. Em 42 (73,7%) instâncias a BTP-ACVO consegue descobrir soluções melhores do que a primeira encontrada. A BTP-ACVO(1^{st}) e a BTP-ACVO falham em 3 instâncias e a HCPLEX falha em 1 instância. A execução HCPLEX apresenta o melhor desvio médio, inclusive quando Média(53) é considerada. Para desvio não superior a 5%, a BTP-ACVO(1^{st}) atingiu 54,4% das instâncias, a BTP-ACVO atingiu 78,9% e a HCPLEX atingiu 93%. A execução BTP-ACVO(1^{st}) encontra 3 soluções ótimas, a BTP-ACVO encontra 8 e a HCPLEX encontra 4.

Para o problema MKP, no teste de Wilcoxon a HCPLEX foi sempre vitoriosa. Entre a HCPLEX e a BTP-ACVO(1^{st}), a HCPLEX é mais rápida em 20 casos, é mais lenta em 9 e ambas gastam o mesmo tempo em 3 casos. Entre a BTP-ACVO(1^{st}) e a BTP-ACVO, no teste de Wilcoxon a BTP-ACVO foi a vencedora. A BTP-ACVO consegue melhorar a primeira solução encontrada para 26

(81,3%) instâncias. A BTP-ACVO(1^{st}) e a HCPLEX não encontram nenhuma solução ótima, já a BTP-ACVO encontra 6 soluções ótimas. A execução HCPLEX apresenta o melhor desvio médio, seguida de perto pela BTP-ACVO. Para desvio não superior a 1%, a BTP-ACVO(1^{st}), a BTP-ACVO e a HCPLEX atingem essa faixa para respectivamente 59,4%, 84,4% e 81,3% das instâncias.

Para o problema MDMKP, o teste de Wilcoxon entre BTP-ACVO(1^{st}) e HCPLEX aponta BTP-ACVO(1^{st}) como vitoriosa. Dentre as 16 instâncias em que ambas conseguem encontrar uma solução inteira-factível, a HCPLEX é mais rápida em 5 casos, é mais lenta em 9 e ambas gastam o mesmo tempo em 2 casos. Naturalmente então, a BTP-ACVO também é vitoriosa no teste de Wilcoxon quando comparada com a HCPLEX. A BTP-ACVO também vence a BTP-ACVO(1^{st}) no teste de Wilcoxon. A BTP-ACVO consegue melhorar a primeira solução encontrada para 25 (78,1%) instâncias. A BTP-ACVO(1^{st}) e a BTP-ACVO falham em 5 instâncias e a HCPLEX falha em 16 instâncias. O melhor desvio médio foi alcançado pela execução BTP-ACVO, resultado que também se repete para Média(16). Para desvio não superior a 5%, a BTP-ACVO(1^{st}) atinge 40,6% das instâncias, a BTP-ACVO atinge 65,6% das instâncias e a HCPLEX 34,4% das instâncias.

Tab. 5.7: GAP – Resultados de curto prazo.

Instâncias		BTP-ACVO(1^{st})			BTP-ACVO			HCPLEX		
Nomes	Melhor Valor	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo
A05100	1698	**1698**	**0,0**	**0**	**1698**	**0,0**	**0**	**1698**	**0,0**	**0**
A05200	3235	3238	0,1	0	**3235**	**0,0**	**0**	**3235**	**0,0**	**0**
A10100	1360	1361	0,1	0	**1360**	**0,0**	**0**	1361	0,1	0
A10200	2623	**2623**	**0,0**	**0**	**2623**	**0,0**	**0**	**2623**	**0,0**	**0**
A20100	1158	**1158**	**0,0**	**0**	**1158**	**0,0**	**0**	1159	0,1	0
A20200	2339	2341	0,1	1	**2339**	**0,0**	**1**	**2339**	**0,0**	**0**
B05100	1843	2111	14,5	1	**1843**	**0,0**	**2443**	–	–	1
B05200	3552	3631	2,2	2	3591	1,1	2555	3590	1,1	1
B10100	1407	1538	9,3	2	1412	0,4	2895	1420	0,9	1
B10200	2827	3028	7,1	5	2870	1,5	700	2926	3,5	4
B20100	1166	1267	8,7	3	1187	1,8	1960	1197	2,7	3
B20200	2339	2393	2,3	4	2356	0,7	1346	2379	1,7	2
C05100	1931	2127	10,2	1	**1931**	**0,0**	**2484**	1995	3,3	4
C05200	3456	3655	5,8	2	3493	1,1	3163	3543	2,5	3
C10100	1402	1605	14,5	2	1482	5,7	65	1520	8,4	9
C10200	2806	3021	7,7	5	2870	2,3	89	2849	1,5	2
C10400	5597	5763	3,0	15	5660	1,1	2601	5803	3,7	4
C15900	11340	11487	1,3	53	11435	0,8	76	11377	0,3	0
C20100	1243	1493	20,1	10	1298	4,4	693	1283	3,2	5
C201600	18802	18982	1,0	379	18975	0,9	460	18891	0,5	0
C20200	2391	2568	7,4	11	2443	2,2	1627	2420	1,2	2
C20400	4782	5090	6,4	45	4853	1,5	3418	4892	2,3	2
C30900	9982	10261	2,8	238	10095	1,1	3542	10072	0,9	1
C401600	17145	17423	1,6	1025	17283	0,8	1858	17212	0,4	0
C40400	4244	4476	5,5	132	4288	1,0	684	4316	1,7	2
C60900	9327	9604	3,0	950	9409	0,9	3167	9410	0,9	1
C801600	16285	16516	1,4	2930	16410	0,8	3568	16395	0,7	1
D05100	6353	6599	3,9	1	6533	2,8	3138	6414	1,0	1
D05200	12742	12932	1,5	2	12885	1,1	2	12771	0,2	0
D10100	6348	6821	7,5	3	6753	6,4	3007	6444	1,5	2
D10200	12432	12879	3,6	4	12835	3,2	5	12535	0,8	1
D10400	24693	25261	2,3	13	25238	2,2	14	25055	1,5	0
D15900	55409	55957	1,0	66	55848	0,8	80	55522	0,2	0
D20100	6190	6716	8,5	3	6625	7,0	16	6347	2,5	3
D201600	97832	98279	0,5	372	98279	0,5	372	97994	0,2	0
D20200	12241	12877	5,2	27	12877	5,2	27	12468	1,9	2
D20400	24574	25176	2,4	53	25149	2,3	54	24792	0,9	1
D30900	54852	55375	1,0	190	55375	1,0	190	55078	0,4	0
D401600	97105	98075	1,0	1266	98004	0,9	1458	97417	0,3	0

Continua na próxima página.

Tab. 5.7 – Continuação da página anterior.

Instâncias		BTP-ACVO(1^{st})			BTP-ACVO			HCPLEX		
Nomes	Melhor Valor	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo
D40400	24392	25351	3,9	227	25262	3,6	251	24717	1,3	1
D60900	54568	55915	2,5	2594	55802	2,3	2774	55014	0,8	1
D801600	97035	–	–	3600	–	–	3600	97516	0,5	0
E05100	12681	13483	6,3	1	12721	0,3	2098	13147	3,7	4
E05200	24930	26396	5,9	2	24977	0,2	3444	25057	0,5	1
E10100	11577	12633	9,1	2	12069	4,2	3139	12725	9,9	9
E10200	23307	26071	11,9	7	24897	6,8	2642	23588	1,2	1
E10400	45746	47209	3,2	12	47112	3,0	3004	46337	1,3	1
E15900	102421	103645	1,2	73	103645	1,2	73	102974	0,5	1
E20100	8436	9630	14,2	13	9630	14,2	13	8965	6,3	7
E201600	180646	182775	1,2	672	182775	1,2	672	181526	0,5	0
E20200	22379	24597	9,9	25	24597	9,9	25	23146	3,4	3
E20400	44877	47125	5,0	51	47125	5,0	51	45411	1,2	1
E30900	100429	103442	3,0	359	103442	3,0	359	101838	1,4	1
E401600	178294	183836	3,1	2108	182929	2,6	2174	179173	0,5	1
E40400	44562	49918	12,0	422	49890	12,0	476	45689	2,5	3
E60900	100153	–	–	3600	–	–	3600	101650	1,5	2
E801600	176828	–	–	3600	–	–	3600	178453	0,9	1
Média			4,9			2,5			1,6	
Média(53)			4,8			2,5			1,7	
Falhas			3			3			1	

Tab. 5.8: GAP – Resultados agregados de curto prazo.

	BTP-ACVO(1^{st})		BTP-ACVO		HCPLEX	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	11	19,3	21	36,8	28	49,1
1% a 5%	20	35,1	24	42,1	25	43,9
5% a 10%	16	28,1	7	12,3	3	5,3
10% a 25%	7	12,3	2	3,5	0	0,0
25% a 50%	0	0,0	0	0,0	0	0,0
50% a 100 %	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0
Falhas	3	5,3	3	5,3	1	1,8

Tab. 5.9: MKP – Resultados de curto prazo.

Instâncias		BTP-ACVO(1^{st})			BTP-ACVO			HCPLEX		
Nomes	Melhor Valor	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo
mkp-5-100-00	24381	24088	1,2	0	**24381**	**0,0**	198	24296	0,3	3
mkp-5-100-01	24274	23443	3,4	0	**24274**	**0,0**	265	24109	0,7	2
mkp-5-100-02	23551	23071	2,0	0	**23551**	**0,0**	332	23518	0,1	3
mkp-5-100-03	23534	22661	3,7	0	**23534**	**0,0**	438	23285	1,1	3
mkp-5-250-00	59312	56359	5,0	0	59215	0,2	3573	58926	0,7	0
mkp-5-250-01	61472	61031	0,7	0	61391	0,1	3124	61310	0,3	0
mkp-5-250-02	62130	60139	3,2	1	61719	0,7	3583	61914	0,3	0
mkp-5-250-03	59463	56432	5,1	0	57858	2,7	79	59311	0,3	0
mkp-5-500-10	218428	218201	0,1	2	218281	0,1	2590	218279	0,1	4
mkp-5-500-11	221202	220672	0,2	2	220895	0,1	3004	221085	0,1	4
mkp-5-500-12	217542	216860	0,3	2	216994	0,3	2283	217424	0,1	5
mkp-5-500-13	223560	223324	0,1	1	223487	0^{+}	2504	223343	0,1	4
mkp-10-100-00	23064	22343	3,1	0	23025	0,2	2535	22891	0,8	0
mkp-10-100-01	22801	22041	3,3	0	22672	0,6	3575	22675	0,6	0
mkp-10-100-02	22131	22081	0,2	0	**22131**	**0,0**	2294	21940	0,9	0
mkp-10-100-03	22772	22694	0,3	0	**22772**	**0,0**	1141	22450	1,4	0
mkp-10-500-10	217377	215336	0,9	1	216001	0,6	2554	217112	0,1	0
mkp-10-500-11	219066	216997	0,9	1	217840	0,6	3007	218755	0,1	0
mkp-10-500-12	217797	217130	0,3	1	217130	0,3	1	217573	0,1	0
mkp-10-500-13	216868	215563	0,6	1	215704	0,5	2449	216630	0,1	0
mkp-30-100-00	21946	20691	5,7	0	21367	2,6	400	21647	1,4	1
mkp-30-100-01	21716	21322	1,8	0	21322	1,8	0	21481	1,1	2
mkp-30-100-02	20754	19882	4,2	0	20385	1,8	2768	20245	2,5	2
mkp-30-100-03	21464	18636	13,2	0	20849	2,9	1043	21244	1,0	2
mkp-30-500-10	218068	216006	0,9	1	216669	0,6	3327	217901	0,1	1
mkp-30-500-11	214626	213743	0,4	1	213743	0,4	1	214328	0,1	1
mkp-30-500-12	215914	215177	0,3	1	215177	0,3	1	215784	0,1	1
mkp-30-500-13	217863	217292	0,3	1	217292	0,3	1	217741	0,1	0
mkp-30-500-20	301656	299980	0,6	1	300565	0,4	3044	301383	0,1	0
mkp-30-500-21	300048	298942	0,4	1	298981	0,4	2271	299821	0,1	0
mkp-30-500-22	305038	304865	0,1	1	304865	0,1	1	304919	0^{+}	0
mkp-30-500-23	302004	301398	0,2	1	301554	0,1	1	301720	0,1	0
Média			2,0			0,6			0,5	

Tab. 5.10: MKP – Resultados agregados de curto prazo.

	BTP-ACVO(1^{st})		BTP-ACVO		HCPLEX	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	19	59,4	27	84,4	26	81,3
1% a 5%	10	31,3	5	15,6	6	18,8
5% a 10%	2	6,3	0	0,0	0	0,0
10% a 25%	1	3,1	0	0,0	0	0,0
25% a 50%	0	0,0	0	0,0	0	0,0
50% a 100 %	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0

Tab. 5.11: MDMKP – Resultados de curto prazo.

Instâncias		BTP-ACVO(1^{st})			BTP-ACVO			HCPLEX		
Nomes	Melhor Valor	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo	Objetivo	Desv	Tempo
mdmkp-05-100-2-0-0	28384	26545	6,5	0	28205	0,6	3496	28148	0,8	2
mdmkp-05-100-2-0-13	67147	63270	5,8	0	**67147**	**0,0**	**1765**	66817	0,5	1
mdmkp-05-100-2-1-14	19614	18631	5,0	0	19558	0,3	2557	19328	1,5	3
mdmkp-05-100-2-1-2	10124	9700	4,2	0	9797	3,2	1081	8371	17,3	25
mdmkp-05-100-5-0-1	26280	24606	6,4	0	**26280**	**0,0**	**539**	26275	0+	1
mdmkp-05-100-5-0-14	53386	52072	2,5	0	**53386**	**0,0**	**898**	48415	9,3	11
mdmkp-05-100-5-1-0	10263	4844	52,8	0	9687	5,6	2033	8989	12,4	20
mdmkp-05-100-5-1-14	17346	14479	16,5	0	17084	1,5	3399	13667	21,2	30
mdmkp-05-500-2-0-11	364332	361046	0,9	2	362202	0,6	3405	363891	0,1	0
mdmkp-05-500-2-0-4	157693	152876	3,1	2	155531	1,4	3395	157084	0,4	1
mdmkp-05-500-2-1-14	113353	111722	1,4	1	111914	1,3	1420	113110	0,2	0
mdmkp-05-500-2-1-3	55413	53975	2,6	1	54046	2,5	345	55115	0,5	1
mdmkp-05-500-5-0-1	145350	144720	0,4	2	144783	0,4	2284	144920	0,3	0
mdmkp-05-500-5-0-4	130928	128819	1,6	1	129640	1,0	1977	117847	10,0	11
mdmkp-05-500-5-1-14	99513	97420	2,1	2	97420	2,1	2	99009	0,5	1
mdmkp-05-500-5-1-8	87420	85374	2,3	1	85385	2,3	1703	86822	0,7	1
mdmkp-30-100-15-0-11	39106	36098	7,7	1375	36643	6,3	1849	–	–	0
mdmkp-30-100-15-0-8	28490	24419	14,3	15	26628	6,5	796	–	–	0
mdmkp-30-100-15-1-12	18519	15016	18,9	1130	15016	18,9	1130	–	–	0
mdmkp-30-100-15-1-4	7118	–	–	–	–	–	–	–	–	0
mdmkp-30-100-30-0-12	31988	–	–	–	–	–	–	–	–	0
mdmkp-30-100-30-0-5	23296	–	–	–	–	–	–	–	–	0
mdmkp-30-100-30-1-12	10751	–	–	–	–	–	–	–	–	0
mdmkp-30-100-30-1-3	3628	–	–	–	–	–	–	–	–	0
mdmkp-30-500-15-0-13	242660	237802	2,0	1	239302	1,4	2307	–	–	1
mdmkp-30-500-15-0-5	159321	153546	3,6	2	155773	2,2	3301	–	–	1
mdmkp-30-500-15-1-11	102997	95963	6,8	2	99765	3,1	3079	–	–	1
mdmkp-30-500-15-1-2	50134	42446	15,3	2	46411	7,4	2679	–	–	1
mdmkp-30-500-30-0-13	214221	210276	1,8	2	211801	1,1	3433	–	–	1
mdmkp-30-500-30-0-7	156343	146649	6,2	3	152810	2,3	3471	–	–	1
mdmkp-30-500-30-1-14	87187	81487	6,5	2	83296	4,5	3309	–	–	1
mdmkp-30-500-30-1-2	48781	27095	44,5	4	44893	8,0	3131	–	–	1
Média			9,0			3,1			4,7	
Média(16)			7,1			1,4			4,7	
Falhas			5			5			16	

Tab. 5.12: MDMKP – Resultados agregados de curto prazo.

	BTP-ACVO(1^{st})		BTP-ACVO		HCPLEX	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	2	6,3	8	25,0	10	31,3
1% a 5%	11	34,4	13	40,6	1	3,1
5% a 10%	8	25,0	5	15,6	2	6,3
10% a 25%	4	12,5	1	3,1	3	9,4
25% a 50%	1	3,1	0	0,0	0	0,0
50% a 100 %	1	3,1	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0
Falhas	5	15,6	5	15,6	16	50,0

5.3 Resultados computacionais de extensões do curto prazo

Nesta subseção são analisadas as extensões aplicáveis ao procedimento de curto prazo da busca tabu paramétrica. A execução BTP-Cortes substitui o uso da memória de curto prazo pelo uso de cortes. A execução BTP-IM-B&C faz uso de *branch-and-cut* para resolver infactibilidade de metas. A execução BTP-RFO envolve relaxação na restrição de função objetivo e a execução BTP-ACVO (*Probing*) usa os testes de *probing*. As Tabelas 5.13 e 5.14 trazem os resultados das instâncias de GAP, as Tabelas 5.15 e 5.16 o resultados das instâncias de MKP e as Tabelas 5.17 e 5.18 os resultados das instâncias de MDMKP.

Para o problema GAP, o teste de Wilcoxon é inconclusivo na comparação da BTP-ACVO tanto com BTP-Cortes quanto com BTP-ACVO (*Probing*). As execuções BTP-IM-B&C e BTP-RFO são dominadas pela BTP-ACVO no teste de Wilcoxon. O desvio médio é melhor para a execução BTP-ACVO (*Probing*), fato que se repete quando Média(50) é considerada. A BTP-ACVO falha em 3 instâncias, a BTP-Cortes em 3, a BTP-IM-B&C em 7, a BTP-RFO em 3 e a BTP-ACVO (*Probing*) não falha em nenhuma instância. A BTP-ACVO encontra solução ótima para 8 instâncias, a BTP-Cortes, BTP-IM-B&C, BTP-RFO e BTP-ACVO (*Probing*) encontram respectivamente 7, 9, 6 e 8 soluções ótimas. Para desvio não superior a 5%, as execuções BTP-ACVO, BTP-Cortes, BTP-IM-B&C, BTP-RFO e BTP-ACVO (*Probing*) atingem essa faixa para respectivamente 78,9%, 80,7%, 68,4%, 73,7% e 89,5%.

Para o problema MKP, as extensões foram dominadas pela BTP-ACVO pelo teste de Wilcoxon, exceto a execução BTP-IM-B&C que apresentou resultado inconclusivo. O desvio médio que apresenta melhor qualidade é o da execução BTP-IM-B&C. Para desvio não superior a 1%, a BTP-ACVO, a BTP-Cortes, a BTP-IM-B&C, a BTP-RFO e a BTP-ACVO (*Probing*) atingem essa faixa para respectivamente 84,4%, 71,9%, 87,5%, 71,9% e 87,5% e o número de soluções ótimas encontradas, também nessa ordem é 6, 3, 5, 1 e 3.

Para o problema MDMKP, as extensões também foram dominadas pela BTP-ACVO pelo teste de Wilcoxon, exceto a execução BTP-IM-B&C que apresentou resultado inconclusivo. A BTP-ACVO encontra 3 soluções ótimas, a BTP-Cortes não encontra nenhuma solução ótima, a BTP-IM-B&C encontra 3 soluções ótimas, a BTP-RFO não encontra nenhuma solução ótima e a BTP-ACVO (*Probing*) encontra 2 soluções ótimas. A BTP-ACVO falha em 5 instâncias, a BTP-Cortes em 6, a BTP-IM-B&C em 3, a BTP-RFO em 5 e a BTP-ACVO (*Probing*) falha em 4 instâncias. O desvio médio para BTP-ACVO é o melhor de todos e quando Média(26) é considerada esse resultado se repete. Para desvio não superior a 5%, a BTP-ACVO, a BTP-Cortes, a BTP-IM-B&C, a BTP-RFO e a BTP-ACVO (*Probing*) atingem essa faixa para respectivamente 65,6%, 59,4%, 65,6%, 53,1% e 65,6%.

Essas análises mostram que para GAP, a variante BTP-Cortes é uma opção ao uso da memória tabu e o uso de *probing* é vantajoso quando o problema possui restrições do tipo $\sum_{j \in J \subset N} x_j = 1$.

Esse tipo de restrição existe na instância disctom e só é resolvida pela BTP quando os testes de *probing* são ativados. A variante BTP-IM-B&C melhora o desvio médio para MKP e reduz o número de falhas para MDMKP.

Tab. 5.13: GAP – Resultados de extensões do curto prazo.

Instâncias	BTP-ACVO		BTP-Cortes		BTP-IM-B&C		BTP-RFO		BTP-ACVO (*Probing*)	
Nomes	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
A05100	**1698**	**0,0**	**1698**	**0,0**	**1698**	**0,0**	**1698**	**0,0**	**1698**	**0,0**
A05200	**3235**	**0,0**	**3235**	**0,0**	**3235**	**0,0**	**3235**	**0,0**	**3235**	**0,0**
A10100	**1360**	**0,0**	**1360**	**0,0**	**1360**	**0,0**	**1360**	**0,0**	**1360**	**0,0**
A10200	**2623**	**0,0**	**2623**	**0,0**	**2623**	**0,0**	**2623**	**0,0**	**2623**	**0,0**
A20100	**1158**	**0,0**	**1158**	**0,0**	**1158**	**0,0**	**1158**	**0,0**	**1158**	**0,0**
A20200	**2339**	**0,0**	**2339**	**0,0**	**2339**	**0,0**	**2339**	**0,0**	**2339**	**0,0**
B05100	**1843**	**0,0**	**1843**	**0,0**	**1843**	**0,0**	1992	8,1	**1843**	**0,0**
B05200	3591	1,1	3590	1,1	3559	0,2	3601	1,4	3585	0,9
B10100	1412	0,4	1421	1,0	**1407**	**0,0**	1435	2,0	1416	0,6
B10200	2870	1,5	2886	2,1	2896	2,4	2893	2,3	2906	2,8
B20100	1187	1,8	1189	2,0	1184	1,5	1189	2,0	1187	1,8
B20200	2356	0,7	2360	0,9	2356	0,7	2353	0,6	2362	1,0
C05100	**1931**	**0,0**	1932	0,1	**1931**	**0,0**	2001	3,6	**1931**	**0,0**
C05200	3493	1,1	3497	1,2	3475	0,5	3517	1,8	3491	1,0
C10100	1482	5,7	1476	5,3	1446	3,1	1487	6,1	1489	6,2
C10200	2870	2,3	2865	2,1	2874	2,4	2869	2,2	2875	2,5
C10400	5660	1,1	5672	1,3	5715	2,1	5667	1,3	5666	1,2
C15900	11435	0,8	11491	1,3	11533	1,7	11455	1,0	11439	0,9
C20100	1298	4,4	1312	5,6	1324	6,5	1318	6,0	1355	9,0
C201600	18975	0,9	18928	0,7	19046	1,3	18934	0,7	18931	0,7
C20200	2443	2,2	2446	2,3	2468	3,2	2450	2,5	2457	2,8
C20400	4853	1,5	4855	1,5	4915	2,8	4861	1,7	4860	1,6
C30900	10095	1,1	10119	1,4	10189	2,1	10115	1,3	10100	1,2
C401600	17283	0,8	17283	0,8	17658	3,0	17290	0,8	17253	0,6
C40400	4288	1,0	4300	1,3	4316	1,7	4288	1,0	4309	1,5
C60900	9409	0,9	9466	1,5	9971	6,9	9435	1,2	9422	1,0
C801600	16410	0,8	16502	1,3	–	–	16502	1,3	16437	0,9
D05100	6533	2,8	6516	2,6	6420	1,1	6567	3,4	6408	0,9
D05200	12885	1,1	12932	1,5	12846	0,8	12885	1,1	12910	1,3
D10100	6753	6,4	6726	6,0	6708	5,7	6821	7,5	6543	3,1
D10200	12835	3,2	12801	3,0	12844	3,3	12835	3,2	12700	2,2
D10400	25238	2,2	25239	2,2	25293	2,4	25238	2,2	25330	2,6
D15900	55848	0,8	55816	0,7	55991	1,1	55848	0,8	55705	0,5
D20100	6625	7,0	6716	8,5	6603	6,7	6625	7,0	6449	4,2
D201600	98279	0,5	98228	0,4	98329	0,5	98248	0,4	98603	0,8
D20200	12877	5,2	12706	3,8	12905	5,4	12877	5,2	12978	6,0
D20400	25149	2,3	25158	2,4	25176	2,4	25149	2,3	24920	1,4
D30900	55375	1,0	55278	0,8	55288	0,8	55375	1,0	55679	1,5
D401600	98004	0,9	97763	0,7	–	–	98004	0,9	97538	0,4
D40400	25262	3,6	25438	4,3	25321	3,8	25262	3,6	24942	2,3
D60900	55802	2,3	56014	2,6	–	–	55858	2,4	55447	1,6
D801600	–	–	–	–	–	–	–	–	97833	0,8
E05100	12721	0,3	12753	0,6	12740	0,5	12979	2,3	12709	0,2
E05200	24977	0,2	25017	0,3	25084	0,6	25336	1,6	25068	0,6
E10100	12069	4,2	12077	4,3	12888	11,3	12587	8,7	12267	6,0
E10200	24897	6,8	24049	3,2	24022	3,1	25184	8,1	24424	4,8
E10400	47112	3,0	47179	3,1	48046	5,0	47209	3,2	47493	3,8
E15900	103645	1,2	104403	1,9	105287	2,8	103645	1,2	103518	1,1
E20100	9630	14,2	10148	20,3	10862	28,8	9630	14,2	9348	10,8
E201600	182775	1,2	182315	0,9	185565	2,7	182775	1,2	182501	1,0
E20200	24597	9,9	24787	10,8	25424	13,6	24597	9,9	23800	6,3
E20400	47125	5,0	47377	5,6	48506	8,1	47125	5,0	46463	3,5
E30900	103442	3,0	103372	2,9	104583	4,1	103442	3,0	101684	1,2
E401600	182929	2,6	182518	2,4	–	–	183621	3,0	180942	1,5
E40400	49890	12,0	50806	14,0	52073	16,9	49254	10,5	46636	4,7
E60900	–	–	–	–	–	–	–	–	102610	2,5
E801600	–	–	–	–	–	–	–	–	179674	1,6
Média		2,5		2,7		3,5		3,0		2,1
Média(50)		2,5		2,7		3,5		3,1		2,2
Falhas		3		3		7		3		0

Tab. 5.14: GAP – Resultados agregados de extensões do curto prazo.

	BTP-ACVO		BTP-Cortes		BTP-IM-B&C		BTP-RFO		BTP-ACVO (*Probing*)	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	21	36,8	19	33,3	17	29,8	13	22,8	22	38,6
1% a 5%	24	42,1	27	47,4	22	38,6	29	50,9	29	50,9
5% a 10%	7	12,3	5	8,8	7	12,3	10	17,5	5	8,8
10% a 25%	2	3,5	3	5,3	3	5,3	2	3,5	1	1,8
25% a 50%	0	0,0	0	0,0	1	1,8	0	0,0	0	0,0
50% a 100 %	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Falhas	3	5,3	3	5,3	7	12,3	3	5,3	0	0,0

Tab. 5.15: MKP – Resultados de extensões do curto prazo.

Instâncias	BTP-ACVO		BTP-Cortes		BTP-IM-B&C		BTP-RFO		BTP-ACVO (*Probing*)	
Nomes	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
mkp-5-100-00	**24381**	**0,0**	**24381**	**0,0**	**24381**	**0,0**	**24381**	**0,0**	**24381**	**0,0**
mkp-5-100-01	**24274**	**0,0**	**24274**	**0,0**	**24274**	**0,0**	24258	0,1	**24274**	**0,0**
mkp-5-100-02	**23551**	**0,0**	**23551**	**0,0**	**23551**	**0,0**	23538	0,1	**23551**	**0,0**
mkp-5-100-03	**23534**	**0,0**	23477	0,2	**23534**	**0,0**	23418	0,5	23497	0,2
mkp-5-250-00	59215	0,2	58303	1,7	59205	0,2	58037	2,1	59012	0,5
mkp-5-250-01	61391	0,1	61141	0,5	**61472**	**0,0**	61037	0,7	61381	0,1
mkp-5-250-02	61719	0,7	61347	1,3	61889	0,4	60833	2,1	61853	0,4
mkp-5-250-03	57858	2,7	58595	1,5	59281	0,3	58358	1,9	59368	0,2
mkp-5-500-10	218281	0,1	218306	0,1	218289	0,1	218292	0,1	218238	0,1
mkp-5-500-11	220895	0,1	220672	0,2	220858	0,2	220870	0,2	220811	0,2
mkp-5-500-12	216994	0,3	216925	0,3	217144	0,2	217225	0,1	217085	0,2
mkp-5-500-13	223487	0+	223324	0,1	223450	0+	223379	0,1	223455	0+
mkp-10-100-00	23025	0,2	22680	1,7	23063	0+	22440	2,7	23052	0,1
mkp-10-100-01	22672	0,6	22264	2,4	22579	1,0	22066	3,2	22642	0,7
mkp-10-100-02	**22131**	**0,0**	22081	0,2	22081	0,2	22081	0,2	22081	0,2
mkp-10-100-03	**22772**	**0,0**	22694	0,3	22707	0,3	22694	0,3	22707	0,3
mkp-10-500-10	216001	0,6	215781	0,7	216178	0,6	216320	0,5	215899	0,7
mkp-10-500-11	217840	0,6	217770	0,6	218079	0,5	217933	0,5	217936	0,5
mkp-10-500-12	217130	0,3	217203	0,3	217130	0,3	217130	0,3	217130	0,3
mkp-10-500-13	215704	0,5	215635	0,6	215673	0,6	216028	0,4	215672	0,6
mkp-30-100-00	21367	2,6	21100	3,9	21359	2,7	20840	5,0	21526	1,9
mkp-30-100-01	21322	1,8	21322	1,8	21350	1,7	21322	1,8	21322	1,8
mkp-30-100-02	20385	1,8	19924	4,0	20271	2,3	19953	3,9	20118	3,1
mkp-30-100-03	20849	2,9	20777	3,2	20901	2,6	20493	4,5	20751	3,3
mkp-30-500-10	216669	0,6	216297	0,8	216912	0,5	216481	0,7	215954	1,0
mkp-30-500-11	213743	0,4	213743	0,4	213743	0,4	213743	0,4	213743	0,4
mkp-30-500-12	215177	0,3	215177	0,3	215177	0,3	215177	0,3	215177	0,3
mkp-30-500-13	217292	0,3	217292	0,3	217292	0,3	217292	0,3	217292	0,3
mkp-30-500-20	300565	0,4	300457	0,4	300368	0,4	300600	0,4	300213	0,5
mkp-30-500-21	298981	0,4	298942	0,4	299065	0,3	298961	0,4	298729	0,4
mkp-30-500-22	304865	0,1	304865	0,1	304865	0,1	304865	0,1	304865	0,1
mkp-30-500-23	301554	0,1	301398	0,2	301554	0,1	301554	0,1	300932	0,4
Média		0,6		0,9		0,5		1,1		0,6

Tab. 5.16: MKP – Resultados agregados de extensões do curto prazo.

	BTP-ACVO		BTP-Cortes		BTP-IM-B&C		BTP-RFO		BTP-ACVO (*Probing*)	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	27	84,4	23	71,9	28	87,5	23	71,9	28	87,5
1% a 5%	5	15,6	9	28,1	4	12,5	8	25,0	4	12,5
5% a 10%	0	0,0	0	0,0	0	0,0	1	3,1	0	0,0
10% a 25%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
25% a 50%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
50% a 100 %	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0

Tab. 5.17: MDMKP – Resultados de extensões do curto prazo.

Instâncias	BTP-ACVO		BTP-Cortes		BTP-IM-B&C		BTP-RFO		BTP-ACVO (*Probing*)	
Nomes	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
mdmkp-05-100-2-0-0	28205	0,6	27624	2,7	28192	0,7	26849	5,4	28223	0,6
mdmkp-05-100-2-0-13	**67147**	**0,0**	66844	0,5	**67147**	**0,0**	66402	1,1	67143	0+
mdmkp-05-100-2-1-14	19558	0,3	18992	3,2	19589	0,1	18791	4,2	19589	0,1
mdmkp-05-100-2-1-2	9797	3,2	9700	4,2	9841	2,8	9700	4,2	9865	2,6
mdmkp-05-100-5-0-1	**26280**	**0,0**	26116	0,6	**26280**	**0,0**	25728	2,1	**26280**	**0,0**
mdmkp-05-100-5-0-14	**53386**	**0,0**	53296	0,2	**53386**	**0,0**	52954	0,8	**53386**	**0,0**
mdmkp-05-100-5-1-0	9687	5,6	8996	12,3	9274	9,6	8747	14,8	9663	5,8
mdmkp-05-100-5-1-14	17084	1,5	16385	5,5	16853	2,8	15925	8,2	17108	1,4
mdmkp-05-500-2-0-11	362202	0,6	361766	0,7	362211	0,6	363155	0,3	361687	0,7
mdmkp-05-500-2-0-4	155531	1,4	155006	1,7	154999	1,7	155046	1,7	154987	1,7
mdmkp-05-500-2-1-14	111914	1,3	111820	1,4	112062	1,1	111960	1,2	111500	1,6
mdmkp-05-500-2-1-3	54046	2,5	54043	2,5	53975	2,6	53975	2,6	53830	2,9
mdmkp-05-500-5-0-1	144783	0,4	144763	0,4	144740	0,4	145000	0,2	144871	0,3
mdmkp-05-500-5-0-4	129640	1,0	129349	1,2	129777	0,9	129634	1,0	129427	1,1
mdmkp-05-500-5-1-14	97420	2,1	97663	1,9	97769	1,8	97576	1,9	97420	2,1
mdmkp-05-500-5-1-8	85385	2,3	85374	2,3	85415	2,3	85758	1,9	85374	2,3
mdmkp-30-100-15-0-11	36643	6,3	–	–	37205	4,9	37145	5,0	37584	3,9
mdmkp-30-100-15-0-8	26628	6,5	26172	8,1	26404	7,3	26362	7,5	25941	8,9
mdmkp-30-100-15-1-12	15016	18,9	14354	22,5	14996	19,0	15016	18,9	15720	15,1
mdmkp-30-100-15-1-4	–	–	–	–	5259	26,1	–	–	5294	25,6
mdmkp-30-100-30-0-12	–	–	–	–	–	–	–	–	–	–
mdmkp-30-100-30-0-5	–	–	–	–	20247	13,1	–	–	–	–
mdmkp-30-100-30-1-12	–	–	–	–	–	–	–	–	–	–
mdmkp-30-100-30-1-3	–	–	–	–	–	–	–	–	–	–
mdmkp-30-500-15-0-13	239302	1,4	238338	1,8	238220	1,8	238433	1,7	238370	1,8
mdmkp-30-500-15-0-5	155773	2,2	155415	2,5	154545	3,0	154749	2,9	154969	2,7
mdmkp-30-500-15-1-11	99765	3,1	98163	4,7	98046	4,8	97289	5,5	98110	4,7
mdmkp-30-500-15-1-2	46411	7,4	44996	10,2	44328	11,6	44005	12,2	45586	9,1
mdmkp-30-500-30-0-13	211801	1,1	210532	1,7	210989	1,5	211023	1,5	210230	1,9
mdmkp-30-500-30-0-7	152810	2,3	151713	3,0	150542	3,7	151121	3,3	152421	2,5
mdmkp-30-500-30-1-14	83296	4,5	82185	5,7	82763	5,1	82156	5,8	82104	5,8
mdmkp-30-500-30-1-2	44893	8,0	41784	14,3	43766	10,3	42120	13,7	44041	9,7
Média		3,1		4,5		4,8		4,8		4,1
Média(26)		3,0		4,5		3,7		4,8		3,3
Falhas		5		6		3		5		4

Tab. 5.18: MDMKP – Resultados agregados de extensões do curto prazo.

	BTP-ACVO		BTP-Cortes		BTP-IM-B&C		BTP-RFO		BTP-ACVO (*Probing*)	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	8	25,0	5	15,6	8	25,0	4	12,5	7	21,9
1% a 5%	13	40,6	14	43,8	13	40,6	13	40,6	14	43,8
5% a 10%	5	15,6	3	9,4	3	9,4	6	18,8	5	15,6
10% a 25%	1	3,1	4	12,5	4	12,5	4	12,5	1	3,1
25% a 50%	0	0,0	0	0,0	1	3,1	0	0,0	1	3,1
50% a 100 %	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Falhas	5	15,6	6	18,8	3	9,4	5	15,6	4	12,5

5.4 Resultados com uso de memória de longo prazo

Nesta subseção são analisadas as estratégias de longo prazo da busca tabu paramétrica. De cada um dos três grupos de combinações de abordagens listadas na Tabela 4.15, a melhor execução é apresentada. As Tabelas 5.19 e 5.20 trazem os resultados das instâncias de GAP. Os melhores resultados são MF_DIV_INT, RK_2–CD_MAX_CC e RK_1–SDCV. Nos resultados com extensões de curto prazo, o uso de testes de *probing* é bastante eficiente para o GAP. Então a combinação da RK_1–SDCV com os testes de *probing* (RK_1–SDCV (*Probing*)) também é apresentada para o GAP. Das 57 instâncias, 9 não foram consideradas nos testes de Wilcoxon quando o uso da matriz de freqüência está ativado porque a tentativa de criação da matriz de freqüência provoca o esgotamento da memória disponível. As instâncias afetadas foram: C401600, C60900, C801600, D401600, D60900, D801600, E401600, E60900 e E801600. As Tabelas 5.21 e 5.22 trazem os resultados das instâncias de MKP. Os melhores resultados são MF_DIV, RK_1–CD_MAX_CC–CI e RK_1–SDCV. Nos resultados de tratamento de penalidade inteira e infactibilidade de metas, o uso de RB é eficiente para o MKP. Então a combinação da RK_1–SDCV com RB (RK_1–SDCV–RB) também é apresentada para o MKP. As Tabelas 5.23 e 5.24 trazem os resultados das instâncias de MDMKP. Os melhores resultados são MF_DIV, RK_1–CD_MAX_SC e RK_1–SDCV. Nos resultados de tratamento de penalidade inteira e infactibilidade de metas, o uso de DM é bastante eficiente para o MDMKP. Então a combinação da RK_1–SDCV com DM (RK_1–SDCV–DM) também é apresentada para o MDMKP.

Para o problema GAP, as melhores variantes foram a RK_1–SDCV e a RK_1–SDCV (*Probing*). Ambas execuções dominam a BTP-ACVO pelo teste de Wilcoxon e a RK_1–SDCV domina a RK_1–SDCV (*Probing*). A BTP-ACVO falha em 3 instâncias, a MF_DIV_INT falha em 11, a RK_2–

CD_MAX_CC falha em 2, a RK_1–SDCV também falha em 2 instâncias e a RK_1–SDCV (*Probing*) tem o mérito de não falhar em nenhuma instância. A execução BTP-ACVO encontra 8 soluções ótimas, a MF_DIV_INT encontra 7, a RK_2–CD_MAX_CC encontra 7, a RK_1–SDCV encontra 9 soluções ótimas e a RK_1–SDCV (*Probing*) também encontra 9 soluções ótimas. Quando o critério de avaliação é o desvio médio ou Média(46), a execução RK_1–SDCV (*Probing*) é a que apresenta melhores resultados. Para desvio não superior a 5%, a BTP-ACVO atinge essa faixa para 78,9% das instâncias, a MF_DIV_INT atinge 70,2%, a RK_2–CD_MAX_CC 82,5%, a RK_1–SDCV atinge 86% e a RK_1–SDCV (*Probing*) atinge 93% das instâncias.

Para o problema MKP, as melhores variantes foram a RK_1–SDCV e a RK_1–SDCV–RB. As execuções RK_1–SDCV e RK_1–SDCV–RB superam a BTP-ACVO pelo teste de Wilcoxon. Na comparação da RK_1–SDCV–RB com a RK_1–SDCV, o teste de Wilcoxon é inconclusivo. A execução MF–DIV encontra a solução ótima para 6 instâncias, a RK_1–CD_MAX_CC–CI encontra para 6 instâncias, a RK_1–SDCV encontra para 13 instâncias e a RK_1–SDCV–RB encontra a solução ótima para 14 instâncias. Os melhores desvios médios ocorrem para RK_1–SDCV e RK_1–SDCV–RB. Para desvio não superior a 1%, a BTP-ACVO atinge essa faixa para 84,4% das instâncias, a MF–DIV para 84,4%, a RK_1–CD_MAX_CC–CI para 93,8%, a RK_1–SDCV para 100% e a RK_1–SDCV–RB também para 100% das instâncias.

Para o problema MDMKP, as melhores variantes foram a RK_1–SDCV e a RK_1–SDCV–DM. As execuções RK_1–SDCV e RK_1–SDCV–DM dominam a BTP-ACVO pelo teste de Wilcoxon, e o teste envolvendo a RK_1–SDCV e a RK_1–SDCV–DM é inconclusivo. A BTP-ACVO encontra a solução ótima para 3 instâncias, a MF–DIV encontra 4 soluções ótimas, a RK_1–CD_MAX_SC encontra 5 soluções ótimas, a RK_1–SDCV encontra 8 soluções ótimas e a RK_1–SDCV–DM também encontra 8 soluções ótimas. A execução RK_1–SDCV não falha para nenhuma instância e as execuções BTP-ACVO, MF–DIV, RK_1–CD_MAX_SC e RK_1–SDCV–DM falham respectivamente em 5, 4, 2 e 1 instâncias. A variante que apresenta o melhor desvio médio é a RK_1–SDCV e quando Média(27) é considerada, RK_1–SDCV–DM passa a ser a melhor execução. Para desvio não superior a 5%, as execuções BTP-ACVO, MF–DIV, RK_1–CD_MAX_SC, RK_1–SDCV e RK_1–SDCV–DM alcançam essa faixa para respectivamente 65,6%, 65,6%, 71,9%, 100% e 93,8%.

As análises evidenciam a supremacia da exploração de variáveis consistentes e fortemente determinadas em relação às demais estratégias de longo prazo. Para os três problemas, GAP, MKP e MDMKP, explorados nesse capítulo a execução RK_1–SDCV sempre foi superior. Para o MKP, a versão RK_1–SDCV que usa a técnica ACVO é competitiva com a RK_1–SDCV–RB. Para a MDMKP a RK_1–SDCV que não falha em nenhuma instância, supera a RK_1–SDCV–DM que falha em uma, favorecendo assim a manutenção da ACVO quando o longo prazo é ativado.

Tab. 5.19: GAP – Resultados de longo prazo.

Instâncias		BTP-ACVO		MF_DIV_INT		*RK*_2–CD_MAX_CC		*RK*_1–SDCV		*RK*_1–SDCV (*Probing*)	
Nomes	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
A05100	1698	**1698**	**0,0**	**1698**	**0,0**	**1698**	**0,0**	**1698**	**0,0**	**1698**	**0,0**
A05200	3235	**3235**	**0,0**	**3235**	**0,0**	**3235**	**0,0**	**3235**	**0,0**	**3235**	**0,0**
A10100	1360	**1360**	**0,0**	**1360**	**0,0**	**1360**	**0,0**	**1360**	**0,0**	**1360**	**0,0**
A10200	2623	**2623**	**0,0**	**2623**	**0,0**	**2623**	**0,0**	**2623**	**0,0**	**2623**	**0,0**
A20100	1158	**1158**	**0,0**	**1158**	**0,0**	**1158**	**0,0**	**1158**	**0,0**	**1158**	**0,0**
A20200	2339	**2339**	**0,0**	**2339**	**0,0**	**2339**	**0,0**	**2339**	**0,0**	**2339**	**0,0**
B05100	1843	**1843**	**0,0**	**1843**	**0,0**	**1843**	**0,0**	**1843**	**0,0**	**1843**	**0,0**
B05200	3552	3591	1,1	3557	0,1	3572	0,6	**3552**	**0,0**	3553	0+
B10100	1407	1412	0,4	1434	1,9	1408	0,1	1408	0,1	**1407**	**0,0**
B10200	2827	2870	1,5	2880	1,9	2870	1,5	2880	1,9	2857	1,1
B20100	1166	1187	1,8	1182	1,4	1183	1,5	1176	0,9	1194	2,4
B20200	2339	2356	0,7	2357	0,8	2353	0,6	2352	0,6	2354	0,6
C05100	1931	**1931**	**0,0**	1932	0,1	1932	0,1	**1931**	**0,0**	**1931**	**0,0**
C05200	3456	3493	1,1	3466	0,3	3465	0,3	3460	0,1	3464	0,2
C10100	1402	1482	5,7	1454	3,7	1440	2,7	1420	1,3	1432	2,1
C10200	2806	2870	2,3	2854	1,7	2859	1,9	2846	1,4	2869	2,2
C10400	5597	5660	1,1	5650	0,9	5654	1,0	5654	1,0	5661	1,1
C15900	11340	11435	0,8	11435	0,8	11435	0,8	11435	0,8	11452	1,0
C20100	1243	1298	4,4	1303	4,8	1282	3,1	1291	3,9	1327	6,8
C201600	18802	18975	0,9	18930	0,7	18929	0,7	18929	0,7	18913	0,6
C20200	2391	2443	2,2	2444	2,2	2444	2,2	2444	2,2	2465	3,1
C20400	4782	4853	1,5	4851	1,4	4851	1,4	4851	1,4	4849	1,4
C30900	9982	10095	1,1	10147	1,7	10098	1,2	10098	1,2	10109	1,3
C401600	17145	17283	0,8	–	–	17290	0,8	17290	0,8	17271	0,7
C40400	4244	4288	1,0	4288	1,0	4288	1,0	4286	1,0	4288	1,0
C60900	9327	9409	0,9	–	–	9399	0,8	9399	0,8	9517	2,0
C801600	16285	16410	0,8	–	–	16382	0,6	16382	0,6	16386	0,6
D05100	6353	6533	2,8	6505	2,4	6510	2,5	6360	0,1	6361	0,1
D05200	12742	12885	1,1	12885	1,1	12885	1,1	12776	0,3	12799	0,4
D10100	6348	6753	6,4	6687	5,3	6687	5,3	6444	1,5	6470	1,9
D10200	12432	12835	3,2	12553	1,0	12835	3,2	12731	2,4	12700	2,2
D10400	24693	25238	2,2	25238	2,2	25238	2,2	25238	2,2	25330	2,6
D15900	55409	55848	0,8	55887	0,9	55887	0,9	55887	0,9	55705	0,5
D20100	6190	6625	7,0	6625	7,0	6625	7,0	6625	7,0	6449	4,2
D201600	97832	98279	0,5	–	–	98279	0,5	98279	0,5	98567	0,8
D20200	12241	12877	5,2	12877	5,2	12877	5,2	12877	5,2	13006	6,2
D20400	24574	25149	2,3	25149	2,3	25149	2,3	25149	2,3	24920	1,4
D30900	54852	55375	1,0	55375	1,0	55375	1,0	55375	1,0	55796	1,7
D401600	97105	98004	0,9	–	–	98004	0,9	98004	0,9	97538	0,4
D40400	24392	25262	3,6	25262	3,6	25262	3,6	25262	3,6	24942	2,3
D60900	54568	55802	2,3	–	–	55734	2,1	55734	2,1	55447	1,6
D801600	97035	–	–	–	–	–	–	–	–	97833	0,8
E05100	12681	12721	0,3	12707	0,2	12699	0,1	12694	0,1	12685	0+
E05200	24930	24977	0,2	24968	0,2	24976	0,2	24931	0,0	24936	0+
E10100	11577	12069	4,2	11940	3,1	12105	4,6	11777	1,7	11836	2,2
E10200	23307	24897	6,8	23953	2,8	24550	5,3	24048	3,2	24058	3,2
E10400	45746	47112	3,0	47112	3,0	47112	3,0	47112	3,0	47405	3,6
E15900	102421	103645	1,2	103645	1,2	103645	1,2	103645	1,2	103518	1,1
E20100	8436	9630	14,2	9384	11,2	9630	14,2	9323	10,5	9290	10,1
E201600	180646	182775	1,2	–	–	182775	1,2	182775	1,2	182501	1,0
E20200	22379	24597	9,9	24580	9,8	24580	9,8	24580	9,8	23800	6,3
E20400	44877	47125	5,0	47008	4,7	47008	4,7	47008	4,7	46463	3,5
E30900	100429	103442	3,0	103442	3,0	103442	3,0	103442	3,0	101684	1,2
E401600	178294	182929	2,6	–	–	181926	2,0	181926	2,0	180838	1,4
E40400	44562	49890	12,0	49890	12,0	49890	12,0	49890	12,0	46636	4,7
E60900	100153	–	–	–	–	114700	14,5	114700	14,5	102610	2,5
E801600	176828	–	–	–	–	–	–	–	–	179674	1,6
Média			2,5		2,4		2,5		2,1		1,7
Média(46)			2,7		2,4		2,5		2,0		1,8
Falhas			3		11		2		2		0

Tab. 5.20: GAP – Resultados agregados de longo prazo.

	BTP-ACVO		MF_DIV_INT		*RK*_2–CD_MAX_CC		*RK*_1–SDCV		*RK*_1–SDCV(*Probing*)	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	21	36,8	19	33,3	24	42,1	28	49,1	24	42,1
1% a 5%	24	42,1	21	36,8	23	40,4	21	36,8	29	50,9
5% a 10%	7	12,3	4	7,0	5	8,8	3	5,3	3	5,3
10% a 25%	2	3,5	2	3,5	3	5,3	3	5,3	1	1,8
25% a 50%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
50% a 100 %	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Falhas	3	5,3	11	19,3	2	3,5	2	3,5	0	0,0

Tab. 5.21: MKP – Resultados de longo prazo.

Instâncias		BTP-ACVO		MF_DIV		*RK*_1–CD_MAX_CC–CI		*RK*_1–SDCV		*RK*_1–SDCV–RB	
Nomes	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
mkp-5-100-00	24381	**24381**	**0,0**	**24381**	**0,0**	**24381**	**0,0**	**24381**	**0,0**	**24381**	**0,0**
mkp-5-100-01	24274	**24274**	**0,0**	**24274**	**0,0**	**24274**	**0,0**	**24274**	**0,0**	**24274**	**0,0**
mkp-5-100-02	23551	**23551**	**0,0**	**23551**	**0,0**	**23551**	**0,0**	**23551**	**0,0**	**23551**	**0,0**
mkp-5-100-03	23534	**23534**	**0,0**	**23534**	**0,0**	**23534**	**0,0**	**23534**	**0,0**	**23534**	**0,0**
mkp-5-250-00	59312	59215	0,2	59179	0,2	59157	0,3	59231	0,1	**59312**	**0,0**
mkp-5-250-01	61472	61391	0,1	61438	0,1	61468	0+	**61472**	**0,0**	61468	0+
mkp-5-250-02	62130	61719	0,7	61806	0,5	61923	0,3	62075	0,1	**62130**	**0,0**
mkp-5-250-03	59463	57858	2,7	59015	0,8	59200	0,4	59420	0,1	59441	0+
mkp-5-500-10	218428	218281	0,1	218331	0,0	218270	0,1	218396	0+	218334	0+
mkp-5-500-11	221202	220895	0,1	221035	0,1	220936	0,1	221089	0,1	220997	0,1
mkp-5-500-12	217542	216994	0,3	217226	0,1	217093	0,2	217435	0+	217475	0+
mkp-5-500-13	223560	223487	0+	223534	0+	223486	0+	223487	0+	223528	0+
mkp-10-100-00	23064	23025	0,2	23063	0+	23059	0+	**23064**	**0,0**	**23064**	**0,0**
mkp-10-100-01	22801	22672	0,6	22539	1,1	22799	0+	**22801**	**0,0**	**22801**	**0,0**
mkp-10-100-02	22131	**22131**	**0,0**	**22131**	**0,0**	**22131**	**0,0**	**22131**	**0,0**	**22131**	**0,0**
mkp-10-100-03	22772	**22772**	**0,0**	**22772**	**0,0**	22694	0,3	**22772**	**0,0**	**22772**	**0,0**
mkp-10-500-10	217377	216001	0,6	216092	0,6	216192	0,5	217034	0,2	217112	0,1
mkp-10-500-11	219066	217840	0,6	217897	0,5	217990	0,5	218810	0,1	218747	0,1
mkp-10-500-12	217797	217130	0,3	217130	0,3	217130	0,3	217500	0,1	217532	0,1
mkp-10-500-13	216868	215704	0,5	215694	0,5	215771	0,5	216631	0,1	216739	0,1
mkp-30-100-00	21946	21367	2,6	21473	2,2	21771	0,8	**21946**	**0,0**	**21946**	**0,0**
mkp-30-100-01	21716	21322	1,8	21322	1,8	**21716**	**0,0**	**21716**	**0,0**	**21716**	**0,0**
mkp-30-100-02	20754	20385	1,8	20279	2,3	20380	1,8	**20754**	**0,0**	**20754**	**0,0**
mkp-30-100-03	21464	20849	2,9	20859	2,8	21160	1,4	**21464**	**0,0**	**21464**	**0,0**
mkp-30-500-10	218068	216669	0,6	216546	0,7	216755	0,6	217821	0,1	217806	0,1
mkp-30-500-11	214626	213743	0,4	213743	0,4	213743	0,4	214125	0,2	213914	0,3
mkp-30-500-12	215914	215177	0,3	215177	0,3	215177	0,3	215562	0,2	215554	0,2
mkp-30-500-13	217863	217292	0,3	217292	0,3	217292	0,3	217741	0,1	217024	0,4
mkp-30-500-20	301656	300565	0,4	300441	0,4	300655	0,3	301178	0,2	301087	0,2
mkp-30-500-21	300048	298981	0,4	298975	0,4	299014	0,3	299703	0,1	299629	0,1
mkp-30-500-22	305038	304865	0,1	304865	0,1	304865	0,1	304865	0,1	304573	0,2
mkp-30-500-23	302004	301554	0,1	301554	0,1	301554	0,1	301723	0,1	301520	0,2
Média			0,6		0,5		0,3		0,1		0,1

Tab. 5.22: MKP – Resultados agregados de longo prazo.

	BTP-ACVO		MF_DIV		*RK*_1–CD_MAX_CC–CI		*RK*_1–SDCV		*RK*_1–SDCV–RB	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	27	84,4	27	84,4	30	93,8	32	100,0	32	100,0
1% a 5%	5	15,6	5	15,6	2	6,3	0	0,0	0	0,0
5% a 10%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
10% a 25%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
25% a 50%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
50% a 100 %	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0

Tab. 5.23: MDMKP – Resultados de longo prazo.

Instâncias		BTP-ACVO		MF_DIV		*RK*_1–CD_MAX_SC		*RK*_1–SDCV		*RK*_1–SDCV–DM	
Nomes	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
mdmkp-05-100-2-0-0	28384	28205	0,6	28222	0,6	**28384**	**0,0**	**28384**	**0,0**	**28384**	**0,0**
mdmkp-05-100-2-0-13	67147	**67147**	**0,0**	**67147**	**0,0**	**67147**	**0,0**	**67147**	**0,0**	**67147**	**0,0**
mdmkp-05-100-2-1-14	19614	19558	0,3	**19614**	**0,0**	**19614**	**0,0**	**19614**	**0,0**	**19614**	**0,0**
mdmkp-05-100-2-1-2	10124	9797	3,2	9715	4,0	10022	1,0	**10124**	**0,0**	**10124**	**0,0**
mdmkp-05-100-5-0-1	26280	**26280**	**0,0**	**26280**	**0,0**	**26280**	**0,0**	**26280**	**0,0**	**26280**	**0,0**
mdmkp-05-100-5-0-14	53386	**53386**	**0,0**	**53386**	**0,0**	**53386**	**0,0**	**53386**	**0,0**	**53386**	**0,0**
mdmkp-05-100-5-1-0	10263	9687	5,6	9541	7,0	9961	2,9	**10263**	**0,0**	**10263**	**0,0**
mdmkp-05-100-5-1-14	17346	17084	1,5	16935	2,4	17016	1,9	**17346**	**0,0**	**17346**	**0,0**
mdmkp-05-500-2-0-11	364332	362202	0,6	362139	0,6	362536	0,5	363996	0,1	363783	0,2
mdmkp-05-500-2-0-4	157693	155531	1,4	155086	1,7	155485	1,4	157205	0,3	157178	0,3
mdmkp-05-500-2-1-14	113353	111914	1,3	111855	1,3	112423	0,8	113168	0,2	112975	0,3
mdmkp-05-500-2-1-3	55413	54046	2,5	53982	2,6	53997	2,6	55057	0,6	55053	0,6
mdmkp-05-500-5-0-1	145350	144783	0,4	144929	0,3	144745	0,4	145079	0,2	145103	0,2
mdmkp-05-500-5-0-4	130928	129640	1,0	129725	0,9	129634	1,0	130817	0,1	127912	2,3
mdmkp-05-500-5-1-14	99513	97420	2,1	97555	2,0	98234	1,3	98602	0,9	98923	0,6
mdmkp-05-500-5-1-8	87420	85385	2,3	85374	2,3	86042	1,6	86856	0,6	86890	0,6
mdmkp-30-100-15-0-11	39106	36643	6,3	36649	6,3	37012	5,4	38807	0,8	38928	0,5
mdmkp-30-100-15-0-8	28490	26628	6,5	26655	6,4	26643	6,5	28490	0,0	28443	0,2
mdmkp-30-100-15-1-12	18519	15016	18,9	15221	17,8	14905	19,5	18519	0,0	18519	0,0
mdmkp-30-100-15-1-4	7118	–	–	–	–	5812	18,3	7118	0,0	6891	3,2
mdmkp-30-100-30-0-12	31988	–	–	–	–	30978	3,2	31087	2,8	31877	0,3
mdmkp-30-100-30-0-5	23296	–	–	21188	9,0	19872	14,7	23080	0,9	22897	1,7
mdmkp-30-100-30-1-12	10751	–	–	–	–	–	–	10751	0,0	10126	5,8
mdmkp-30-100-30-1-3	3628	–	–	–	–	–	–	3628	0,0	–	–
mdmkp-30-500-15-0-13	242660	239302	1,4	239240	1,4	239563	1,3	240841	0,7	241186	0,6
mdmkp-30-500-15-0-5	159321	155773	2,2	155450	2,4	156566	1,7	158546	0,5	158398	0,6
mdmkp-30-500-15-1-11	102997	99765	3,1	99799	3,1	100274	2,6	102546	0,4	102223	0,8
mdmkp-30-500-15-1-2	50134	46411	7,4	45725	8,8	46587	7,1	48580	3,1	48824	2,6
mdmkp-30-500-30-0-13	214221	211801	1,1	211575	1,2	211876	1,1	213704	0,2	214221	0,0
mdmkp-30-500-30-0-7	156343	152810	2,3	153062	2,1	155251	0,7	155071	0,8	156343	0,0
mdmkp-30-500-30-1-14	87187	83296	4,5	83891	3,8	83732	4,0	85805	1,6	86888	0,3
mdmkp-30-500-30-1-2	48781	44893	8,0	44859	8,0	44772	8,2	47474	2,7	48591	0,4
Média			3,1		3,4		3,7		0,6		0,7
Média(27)			3,1		3,2		2,7		0,5		0,4
Falhas			5		4		2		0		1

Tab. 5.24: MDMKP – Resultados agregados de longo prazo.

	BTP-ACVO		MF_DIV		*RK*_1–CD_MAX_SC		*RK*_1–SDCV		RK1–SDCV–DM	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	8	25,0	8	25,0	10	31,3	28	87,5	26	81,3
1% a 5%	13	40,6	13	40,6	13	40,6	4	12,5	4	12,5
5% a 10%	5	15,6	6	18,8	4	12,5	0	0,0	1	3,1
10% a 25%	1	3,1	1	3,1	3	9,4	0	0,0	0	0,0
25% a 50%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
50% a 100 %	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0	0	0,0	0	0,0
Falhas	5	15,6	4	12,5	2	6,3	0	0,0	1	3,1

5.5 O problema MDMKP resolvido pelo Cplex e pela BTP

As execuções da BTP em instâncias do problema da mochila multidimensional com restrições de demanda foram bastante promissoras. Em muitos casos a BTP conseguiu encontrar soluções inteiras-factíveis de melhor qualidade do que aquelas encontradas pelo Cplex. Então esta seção apresenta os resultados do *branch-and-cut* do Cplex e os compara com os melhores resultados encontrados pela BTP. As Tabelas 5.25 e 5.26 contêm os resultados. A execução CPLEX–2h é relativa ao *branch-and-cut* do Cplex com limite de tempo de 2 horas. As execuções *RK*_1–SDCV e *RK*_1–SDCV–2h são relativas à BTP de referência com a identificação de variáveis consistentes e fortemente determinadas com limites de tempo de 1 hora e 2 horas, respectivamente.

O teste de Wilcoxon envolvendo as execuções CPLEX–2h e *RK*_1–SDCV mostrou-se inconclusivo. A execução CPLEX–2h vence em 15 casos, perde em 9 e ocorrem 8 empates. Também é inconclusivo o teste de Wilcoxon envolvendo as execuções CPLEX–2h e *RK*_1–SDCV–2h. A execução CPLEX–2h vence em 13 casos, perde em 11 e ocorrem 8 empates. A execução *RK*_1–SDCV–2h supera a execução *RK*_1–SDCV pelo teste de Wilcoxon. A *RK*_1–SDCV–2h encontra soluções ainda melhores para 16 instâncias. A execução CPLEX–2h falha em 4 instâncias, e as execuções *RK*_1–SDCV e *RK*_1–SDCV–2h não falham em nenhuma instância. As três execuções encontram a solução ótima para 8 instâncias. Os desvios médios para CPLEX–2h, *RK*_1–SDCV e *RK*_1–SDCV–2h são respectivamente 0,9%, 0,6% e 0,4%. Quando Média(28) é considerada, os desvios são 0,9%, 0,5% e 0,3%. A execução *RK*_1–SDCV–2h melhorou ainda mais o "Melhor Valor" das instâncias mdmkp-30-500-30-0-13, mdmkp-30-500-30-0-7 e mdmkp-30-500-30-1-2. Para desvio não superior a 5%, a execução CPLEX–2h atinge essa faixa para 84,4% das instâncias e as execuções *RK*_1–SDCV e *RK*_1–SDCV–2h atingem essa faixa para 100% das instâncias.

Tab. 5.25: Execuções de MDMKP pelo Cplex e pela BTP.

Instâncias		CPLEX–2h		*RK*_1–SDCV		*RK*_1–SDCV–2h	
Nomes	Melhor Valor	Objetivo	Desv	Objetivo	Desv	Objetivo	Desv
mdmkp-05-100-2-0-0	28384	**28384**	**0,0**	**28384**	**0,0**	**28384**	**0,0**
mdmkp-05-100-2-0-13	67147	**67147**	**0,0**	**67147**	**0,0**	**67147**	**0,0**
mdmkp-05-100-2-1-14	19614	**19614**	**0,0**	**19614**	**0,0**	**19614**	**0,0**
mdmkp-05-100-2-1-2	10124	**10124**	**0,0**	**10124**	**0,0**	**10124**	**0,0**
mdmkp-05-100-5-0-1	26280	**26280**	**0,0**	**26280**	**0,0**	**26280**	**0,0**
mdmkp-05-100-5-0-14	53386	**53386**	**0,0**	**53386**	**0,0**	**53386**	**0,0**
mdmkp-05-100-5-1-0	10263	**10263**	**0,0**	**10263**	**0,0**	**10263**	**0,0**
mdmkp-05-100-5-1-14	17346	**17346**	**0,0**	**17346**	**0,0**	**17346**	**0,0**
mdmkp-05-500-2-0-11	*364332*	364332	0,0	363996	0,1	364102	0,1
mdmkp-05-500-2-0-4	*157693*	157693	0,0	157205	0,3	157218	0,3
mdmkp-05-500-2-1-14	*113353*	113353	0,0	113168	0,2	113172	0,2
mdmkp-05-500-2-1-3	*55413*	55413	0,0	55057	0,6	55262	0,3
mdmkp-05-500-5-0-1	*145350*	145350	0,0	145079	0,2	145272	0,1
mdmkp-05-500-5-0-4	*130928*	130928	0,0	130817	0,1	130889	0,0
mdmkp-05-500-5-1-14	*99513*	99513	0,0	98602	0,9	99044	0,5
mdmkp-05-500-5-1-8	*87420*	87420	0,0	86856	0,6	87095	0,4
mdmkp-30-100-15-0-11	**39106**	38506	1,5	38807	0,8	38832	0,7
mdmkp-30-100-15-0-8	**28490**	27841	2,3	28490	0,0	28490	0,0
mdmkp-30-100-15-1-12	**18519**	17784	4,0	18519	0,0	18519	0,0
mdmkp-30-100-15-1-4	**7118**	5963	16,2	7118	0,0	7118	0,0
mdmkp-30-100-30-0-12	**31988**	–	–	31087	2,8	31087	2,8
mdmkp-30-100-30-0-5	**23296**	–	–	23080	0,9	23143	0,7
mdmkp-30-100-30-1-12	**10751**	–	–	10751	0,0	10751	0,0
mdmkp-30-100-30-1-3	**3628**	–	–	3628	0,0	3628	0,0
mdmkp-30-500-15-0-13	*242660*	242660	0,0	240841	0,7	241294	0,6
mdmkp-30-500-15-0-5	*159321*	159321	0,0	158546	0,5	158760	0,4
mdmkp-30-500-15-1-11	*102997*	102997	0,0	102546	0,4	102546	0,4
mdmkp-30-500-15-1-2	*50134*	50134	0,0	48580	3,1	48580	3,1
mdmkp-30-500-30-0-13	**214592**	212296	1,1	213704	0,4	214592	0,0
mdmkp-30-500-30-0-7	**156797**	155537	0,8	155071	1,1	156797	0,0
mdmkp-30-500-30-1-14	*87187*	87187	0,0	85805	1,6	86146	1,2
mdmkp-30-500-30-1-2	**48976**	48781	0,4	47474	3,1	48976	0,0
Média			0,9		0,6		0,4
Média(28)			0,9		0,5		0,3
Falhas			4		0		0

Tab. 5.26: Resultados agregados das execuções de MDMKP pelo Cplex e pela BTP.

	CPLEX–2h		RK_1–SDCV		RK_1-SDCV–2h	
Porcentagens agregadas	Qtde	(%)	Qtde	(%)	Qtde	(%)
Ótimas ou até 1%	23	71,9	27	84,4	29	90,6
1% a 5%	4	12,5	5	15,6	3	9,4
5% a 10%	0	0,0	0	0,0	0	0,0
10% a 25%	1	3,1	0	0,0	0	0,0
25% a 50%	0	0,0	0	0,0	0	0,0
50% a 100 %	0	0,0	0	0,0	0	0,0
Acima de 100%	0	0,0	0	0,0	0	0,0
Falhas	4	12,5	0	0,0	0	0,0

Capítulo 6

Conclusões e Trabalhos Futuros

6.1 Conclusões

Neste trabalho foi realizado um estudo computacional da busca tabu paramétrica para problemas de programação inteira mista 0–1. Nessa abordagem paramétrica, inequações de ramificação associadas com as variáveis binárias, conhecidas como condições de metas, são impostas indiretamente pela adição de penalidades na função objetivo da relaxação linear do PIM. Esse problema relaxado e penalizado é chamado de PL'. As variáveis binárias do PIM, que na solução ótima do PL' assumem valores fracionários, são inseridas em conjuntos apropriados baseado em medidas de classificação. Cada variável nesses conjuntos recebe uma resposta adequada no intuito da busca encontrar uma solução inteira-factível. As variáveis na solução ótima do PL' que violam as suas metas definem o conjunto G das variáveis com metas infactíveis. Uma medida chamada resistência à meta é usada para selecionar o conjunto de variáveis $x_j, j \in G_p \subseteq G$, que recebem novas metas no sentido oposto e o conjunto de variáveis $x_j, j \in G_s \subseteq G - G_p$ representam as variáveis que são liberadas de suas metas. O conjunto das variáveis potencialmente infactíveis é composto das variáveis com metas satisfeitas, mas que se tiverem as suas penalidades diminuídas em uma pequena quantidade, tornam-se infactíveis em relação às suas metas. As medidas de resistência a metas das variáveis potencialmente infactíveis sempre são classificadas como menores em relação às variáveis com metas infactíveis. As mesmas respostas das variáveis com meta infactíveis se aplicam às variáveis potencialmente infactíveis. Sendo assim, dependendo da medida de resistência a metas, podem ser incluídas no conjunto G_p ou no conjunto G_s. O conjunto D contém as variáveis binárias sem metas associadas, mas que são fracionárias na solução ótima do PL'. Uma medida de escolha preferencial baseada nas penalidades inteiras implicadas ao forçar uma variável a assumir o valor zero ou um é definida. As variáveis com maiores valores de escolha preferencial fazem parte do conjunto $D_0 \in D$ e recebem metas baseadas nas penalidades inteiras. Uma restrição tabu é associada a uma resposta ao proibi-la de ser executada

se a resposta na direção oposta foi executada em alguma iteração recente da busca. Há uma restrição que controla a qualidade da função objetivo, e sempre é atualizada quando uma nova solução inteira-factível é encontrada, tornando essa solução infactível para o PL'.

Os testes computacionais foram conduzidos em 219 instâncias, das quais 98 fazem da Miplib 2003 e Mittelmann (2003), 57 são instâncias de problema de designação generalizada (GAP), 32 são instâncias de problema da mochila multidimensional (MDK) e 32 são instâncias de problema de mochila multidimensional com restrições de demanda (MDMKP).

Três estratégias de medida de resistência a metas e penalidade inteira foram avaliadas: distância a meta (DM), *reliability branching* (RB) e *active-constraint variable ordering* (ACVO). O teste não-paramétrico de classificação com sinais de Wilcoxon mostra a superioridade da técnica ACVO. O uso de memória para o peso associado com uma meta estabelecida com queda exponencial com o avanço das iterações é melhor do que o uso de penalidades constantes. O uso de uma estratégia adaptativa que define a cardinalidade dos conjuntos G_p, G_s e D_0 de acordo com a região sendo visitada é mais eficiente do que o uso de uma estratégia estática que extrai frações fixas dos conjuntos G e D. A BTP de curto prazo (BTP-ACVO) e a versão que é encerrada quando a primeira solução inteira-factível é encontrada (BTP-ACVO(1^{st})) foram comparadas com as heurísticas do Cplex (HCPLEX) e com a *Objective Feasibility Pump* (OFP) com limite de tempo de 3600s. Nos problemas da Miplib, o teste de Wilcoxon não mostra uma dominância entre a OFP, HCPLEX e BTP-ACVO(1^{st}), mas a BTP-ACVO supera as demais pelo teste de Wilcoxon, mostrando a habilidade da busca tabu paramétrica em encontrar outras soluções de melhor qualidade quando mais tempo computacional é oferecido.

Foram propostas algumas extensões de curto prazo como relaxação na restrição de função objetivo, uso de cortes derivados do próprio PL' e uso de *branch and cut* para resolver infactibilidades de metas. Dentre elas, a mais promissora foi o uso de *branch-and-cut* para resolver infactibilidades de metas.

As estratégias de longo prazo implementadas são baseadas em conjuntos de referência de soluções. A inclusão de uma solução no conjunto de referência depende da qualidade da solução e de uma medida de dispersão que evita que os elementos do conjunto sejam muito próximos entre si. Como a busca tabu paramétrica encontra poucas soluções inteiras-factíveis, o conjunto de referência aceita soluções com infactibilidade inteira que são penalizadas. Uma matriz de freqüência é construída a partir do conjunto de referência e permite a redefinição dos cálculos de resistência a metas e penalidade inteira. Ao adotar um modo diferente de calcular a resistência a metas e infactibilidade inteira daquele usado no curto prazo, fases de intensificação e diversificação são então conduzidas. Técnicas derivadas de *scatter search* são usadas para gerar centros de subconjuntos do conjunto de referência. Esses centros redefinem novos PL's e dependendo de como são gerados, podem provocar intensificação ou diversificação. Finalmente, um algoritmo que explora variáveis consistentes e

fortemente determinadas é proposto. Progressivamente, variáveis binárias que apresentam consistentemente o mesmo valor no conjunto de referência e que apresentam grande degradação na função objetivo ou infactibilidade caso sejam alteradas, são fixadas com a evolução da busca. Cada uma dessas estratégias apresenta algum resultado positivo para certos grupos de instâncias. Já a exploração de variáveis consistentes e fortemente determinadas apresenta melhoria para todos os subconjuntos de instâncias considerados. Particularmente em instâncias de MDMKP foi capaz de encontrar soluções inteiras-factíveis para as quais o *branch-and-cut* do Cplex 10 falhou.

6.2 Trabalhos futuros

A busca tabu paramétrica utiliza apenas a informação da relaxação linear do PIM para encontrar uma solução inteira-factível. As heurísticas de busca local como ramificação local, *relaxation induced neighborhood search* (RINS) e *distance induced neighborhood search* (DINS), fazem uso intensivo de *branch-and-cut* para melhorar uma solução conhecida em torno de uma vizinhança definida. Então, a BTP que explora apenas a relaxação linear do problema, não obteve sucesso quando comparada com as heurísticas de busca local.

Um modo de melhorar a qualidade das soluções encontradas pela BTP pode ser alcançado com o acoplamento de pacotes comerciais e uso conjunto de *branch-and-cut*. A primeira abordagem a ser usada seria o uso de uma heurística de busca local quando uma solução inteira-factível é encontrada pela BTP. A candidata natural a ser aplicada é a RINS, por ser a mais fácil de ser implementada e por ter apresentado resultados superiores à ramificação local. A segunda abordagem é um pouco mais sofisticada e deve ser inserida no algoritmo Coordenador da Figura 3.12. Depois que o algoritmo Coordenador invoca o procedimento FixaVars, o *branch-and-cut* é ativado por um limite de tempo. Esse tempo precisa ser determinado empiricamente bem como a profundidade máxima da árvore apresentada na Figura 3.10, que provavelmente será reduzida.

Referências Bibliográficas

[1] Achterberg, T., Berthold, T.: Improving the Feasibility Pump, Discrete Optimization **4**, 77–86 (2007).

[2] Achterberg, T., Koch, T. , Martin, A.: Branching Rules Revisited, Operations Research Letters **33**, 42–54 (2005).

[3] Achterberg, T., Koch, T. , Martin, A.: Miplib 2003, Operations Research Letters **34** (4), 1–12 (2006).

[4] Arntzen, H., Hvattum, L. M., Løkketangen A.: Adaptive memory search for multidemand multidimensional knapsack problems, Computers and Operations Research **33**, 2508–2525 (2006).

[5] Atamtürk, A., Savelsbergh, M. W. P.: Integer-programming software systems, Annals of Operations Research **140**, 67–124 (2005).

[6] Balas, E.: Integer programming and convex analysis: intersection cuts from outer polars, Mathematical Programming **2**, 330–382 (1972).

[7] Balas, E., Bowman, V.J., Glover, F., Sommer, D.: An intersection cut from the dual of the unit hypercube, Operations Research **19**, 40–44 (1971).

[8] Balas, E., Ceria, S., Dawande, M., Margot, F., Pataki, G.: OCTANE: A new heuristic for pure 0–1 programs, Operations Research **49** (2), 207–225 (2001).

[9] Balas, E., Jeroslow, R.: Canonical cuts on the unit hypercube, SIAM Journal on Applied Mathematics **23** (1) 61–69 (1972).

[10] Balas, E., Martin, C.H.: Pivot-and-complement: A heuristic for 0–1 programming, Management Science **26** (1), 86–96 (1980).

[11] Balas, E., Schmieta, S., Wallace, C.: Pivot and shift–a mixed integer programming heuristic, Discrete Optimization **1** (1), 3–12 (2004).

[12] Beasley, J. E.: OR–Library. Distributing test problems by electronic mail, The Journal of the Operational Research Society **41** (11), 1069–1072 (1990).

[13] Bénichou, M., Gauthier, J.M., Girodet, P., Hentges, G., Ribière, G., Vincent, O.: Experiments in mixed-integer linear programming, Mathematical Programming **1**, 76–94 (1971).

[14] Bertacco, L., Fischetti, M., Lodi, A.: A feasibility pump heuristic for general mixed-integer problems, Discrete Optimization **4**, 63–76 (2007).

[15] Bixby, R. E, Ceria, S., McZeal, C. M., Savelsbergh, M. W. P.: An updated mixed integer programming library: MIPLIB 3, Technical Report 98–03 (1998).

[16] Cappanera, P., Trubian, M.: A local-search-based heuristic for the demand-constrained multidimensional knapsack problem, Informs Journal on Computing **17** (1), 82–98 (2005).

[17] Chu, P. C., Beasley, J. E.: A genetic algorithm for the generalized assignment problem, Computers and Operations Research **24**, 17–23 (1997).

[18] Chu, P. C., Beasley, J. E.: A genetic algorithm for the multidimensional knapsack problem, Journal of Heuristics **4**, 63–86 (1998).

[19] COIN-OR. *http://www.coin-or.org*, (2009).

[20] Danna, E., Rothberg, E., Le Pape, C.: Exploring Relaxation Induced Neighborhoods to Improve MIP Solutions, Mathematical Programming Series A **102** (1), 71–90 (2005).

[21] Dash Optimization, XPRESS. *http://www.dashoptimization.com*, (2009).

[22] Eckstein, J.: Parallel branch-and-bound algorithms for general mixed integer programming on the CM-5, SIAM Journal on Optimization **4** 794–814 (1994).

[23] Eckstein, J., Nediak, M.: Pivot, cut, and dive: A heuristic for 0–1 mixed integer programming, Journal of Heuristics **13**, 471–503 (2007).

[24] Faaland, B. H., Hillier, F. S.: Interior path methods for heuristic integer programming procedures, Operations Research **27** (6), 1069–1087 (1979).

[25] Feo, T.A., Resende, M. G. C.: Greedy randomized adaptive search procedures, Journal of Global Optimization **6**, 109–133 (1995).

[26] Fischetti, M., Glover, F., Lodi, A.: The Feasibility Pump, Mathematical Programming Series A **104** (1), 91–104 (2005).

[27] Fischetti, M., Lodi, A.: Local Branching, Mathematical Programming Series B **98**, 23–47 (2003).

[28] Fischetti, M., Lodi, A.: Repairing MIP infeasibility through local branching, Computers and Operations Research **35**, 1436–1445 (2008).

[29] Glover, F.: Convexity cuts and cut search, Operations Research **21**, 123–134 (1973).

[30] Glover, F.: Parametric Branch and Bound, OMEGA, The International Journal of Management Science **6** (2), 145–152 (1978).

[31] Glover, F.: Scatter Search and Star-Paths: Beyond the Genetic Metaphor, OR Spectrum **17**, 125–137 (1995).

[32] Glover, F.: A template for scatter search and path relinking, Lecture Notes In Computer Science **1363**, 3–54 (1997).

[33] Glover, F.: Parametric tabu-search for mixed integer programs, Computers and Operations Research **33**, 2449–2494 (2006).

[34] Glover, F.: Infeasible/feasible search trajectories and directional rounding in integer programming, Journal of Heuristics **13**, 505–541 (2007).

[35] Glover, F., Laguna, M.: General purpose heuristics for integer programming –part I, Journal of Heuristics **2**, 343–358 (1997).

[36] Glover, F., Laguna, M.: General purpose heuristics for integer programming –part II, Journal of Heuristics **3**, 161–179 (1997).

[37] Glover, F., Laguna, M.: Tabu Search. Kluwer Academic Publisher, Boston, Dordrecht, London, 1997.

[38] Glover, F., Laguna, M., Martí: Fundamentals of scatter search and path relinking, Control and Cybernetics **29** (3), 653–684 (2000).

[39] Glover, F., Tangedahl, L.: Dynamic strategies for branch and bound, OMEGA, The International Journal of Management Science **4** (5), 571–576 (1976).

[40] Ghosh, S.: DINS, a MIP improvement heuristic, In Proceedings of 12*th* Integer Programming and Combinatorial Optimization – Lecture Notes in Computer Science **4513**, 310–323 (2007).

[41] Goldberg, D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Professional, 1989.

[42] Golden, B. L., Stewart, W. R.: Empirical Analysis of Heuristics in The Traveling Salesman Problem. E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys, eds., John Wiley, 207–250 (1985).

[43] Gomory, R. E.: An algorithm for the mixed integer problem, Research Memorandum RM-2597, Rand Corporation, Santa Monica (1960). *Apud* Glover, F. , Laguna, M.: General purpose heuristics for integer programming –part I, Journal of Heuristics **3**, 161–179 (1997).

[44] Hanafi, S., Glover, F.: Resolution search and dynamic branch-and-bound, Journal of Combinatorial Optimization **6** (4), 401–423 (2002).

[45] Hansen, P. , Mladenović, N., Urošević, D.: Variable neighborhood search and local branching, Computers and Operations Research **33** 3034–3045 (2006).

[46] Hillier, F. S.: Efficient heuristic procedures for integer linear programming with an interior, Operations Research **17**, 600–637 (1969).

[47] Ibaraki, T., Ohashi, T., Mine, H.: A heuristic algorithm for mixed-integer programming problems, Mathematical Programming Study **2**, 115–136 (1974).

[48] ILOG, Cplex. *http://www.ilog.com/products/cplex*, (2009).

[49] Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P.: Progress in linear programming-based algorithms for integer programming: an exposition, Informs Journal on Computing **12** (1), 2–23 (2000).

[50] Van Laarhoven, P. J. M., Aarts, E. H. L.: Simulated Annealing: Theory and Practice. Kluwer Academic Publishers, 1987.

[51] Linderoth, J.T., Savelsbergh, M.W.P.: A computational study of search strategies for mixed integer programming, Informs Journal on Computing **11**, 173–187 (1999).

[52] Løkketangen, A., Glover, F.: Solving zero/one mixed integer programming problems using tabu search, European Journal of Operations Research **106**, 624–658 (1998).

[53] Løkketangen, A., Jörnsten, K., Storøy, S.: Tabu search within a pivot and complement framework, International Transactions in Operational Research **1** (3), 305–316 (1994).

[54] Mittelmann, H.: Decision tree for optimization software: Benchmarks for optimization software, *http://plato.asu.edu/bench.html*, (2003).

[55] Mladenović, N., Hansen, P.: Variable neighborhood search, Computers and Operations Research **24**, 1097–1100 (1997).

[56] Mosteller, F., Rourke, R. E. K.: Sturdy statistics: nonparametrics and order statistics. Addison-Wesley, 1973.

[57] Patel, J., Chinneck, J. W.: Active constraint variable ordering for faster feasibility of mixed integer linear programs, Mathematical Programming Series A **110**, 445–474 (2007).

[58] Petersen, C. C.: Computational experience with variants of the Balas algorithm applied to the selection of R&D projects, Management Science **13**, 736–750 (1967). *Apud* Løkketangen, A., Jörnsten, K., Storøy, S.: Tabu search within a pivot and complement framework, International Transactions in Operational Research **1** (3), 305–316 (1994).

[59] Saltzman, R. M., Hillier, F. S.: A heuristic ceiling point algorithm for general integer linear programming, Management Science **38** (2), 263–283 (1992).

[60] Santos, R. F.: Uso de cortes canônicos no método de ramificação local para problemas inteiros 0–1 mistos, Dissertação (Mestrado em Ciência da Computação), Instituto de Computação, Universidade Estadual de Campinas, SP, 2006.

[61] Savelsbergh, M. W. P.: Preprocessing and probing techniques for mixed integer programming problems, ORSA Journal on Computing **6** (2), 1445–1454 (1994).

[62] Tomlin, J. A.: An improved branch-and-bound method for integer programming, Operations Research **19**, 1070–1075 (1971).

[63] Vanderbei, R. J.: Linear Programming: Foundations and Extensions – Second Edition. Kluwer Academic Publishers, 1997.

[64] Yagiura, M., Ibaraki, T., Glover, F.: A path relinking approach with ejection chains for the generalized assignment problem, European Journal of Operational Research **169**, 548–569 (2006).

Apêndice A

Técnicas Elementares de *Probing*

Essa discussão considera apenas problemas de programação inteira pura do tipo 0–1 de acordo com o modelo a seguir:

$$\text{(PI) } z = \min cx \tag{A.1}$$

$$\text{sujeito a}$$

$$l \leq Ax \leq u \tag{A.2}$$

$$x_j \in \{0,1\} \quad \forall j \in \{1,\ldots,n\} \tag{A.3}$$

no qual A é uma matrix $(m \times n)$ enquanto $l^t = [rl_1,\ldots,rl_m]$ e $u^t = [ru_1,\ldots,ru_m]$ são os limitantes inferiores e superiores das restrições. Os elementos α_j^i são os coeficientes na matriz A e $a_j^i = \left|\alpha_j^i\right|$. Para a análise de cada restrição i, o conjunto de índices das variáveis binárias $\mathcal{B} = \{1,\ldots,n\}$ é particionado em $\mathcal{B}^+ = \left\{j \in \mathcal{B} \,|\, \alpha_j^i > 0\right\}$ e $\mathcal{B}^- = \left\{j \in \mathcal{B} \,|\, \alpha_j^i < 0\right\}$.

A.1 Teste da infactibilidade para uma restrição

Cada restrição em (A.2) pode ser reescrita na forma da expressão (A.4).

$$rl_i \leq -\sum_{j\in\mathcal{B}^-} a_j^i x_j + \sum_{j\in\mathcal{B}^+} a_j^i x_j \leq ru_i \tag{A.4}$$

Num processo de busca, eventualmente algumas variáveis estarão fixadas e então alguns ajustes precisam ser feitos em rl_i e ru_i. Todos os $x_j \in \mathcal{B}^- \cup \mathcal{B}^+$ e que assumem o valor zero podem ser ignorados pois não exigem atualizações nos limitantes inferiores e superiores. Porém, para todo $x_j \in \mathcal{B}^- \cup \mathcal{B}^+$ com valor fixo em um, atualizam-se os limitantes superiores e inferiores das restrições fazendo $rl_i \leftarrow rl_i - \sum_{j\in\mathcal{B}^+|x_j=1} a_j^i + \sum_{j\in\mathcal{B}^-|x_j=1} a_j^i$ e $ru_i \leftarrow ru_i - \sum_{j\in\mathcal{B}^+|x_j=1} a_j^i + \sum_{j\in\mathcal{B}^-|x_j=1} a_j^i$. Cada

restrição i pode ser testada quanto à factibilidade por uma análise dos limitantes inferior e superior da restrição. Sejam os dois PLs (A.5) e (A.6) que podem ser facilmente resolvidos por simples inspeção, resultando em $L^i_{\min} = -\sum_{j \in \mathcal{B}^-} a^i_j$ e $U^i_{\max} = \sum_{j \in \mathcal{B}^+} a^i_j$.

$$L^i_{\min} = \min \sum_{j \in \mathcal{B}^+} a^i_j x_j - \sum_{j \in \mathcal{B}^-} a^i_j x_j \tag{A.5}$$

sujeito a

$$x_j \in \{0,1\} \quad \forall j \in \{1,\ldots,n\}$$

e

$$U^i_{\max} = \max \sum_{j \in \mathcal{B}^+} a^i_j x_j - \sum_{j \in \mathcal{B}^-} a^i_j x_j \tag{A.6}$$

sujeito a

$$x_j \in \{0,1\} \quad \forall j \in \{1,\ldots,n\}$$

Se $L^i_{\min} > ru_i \Rightarrow -\sum_{j \in \mathcal{B}^-} a^i_j > ru_i$ então o problema é infactível conforme ilustra a Figura A.1.

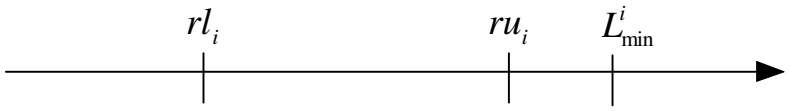

Fig. A.1: Limitante inferior superior ao *upper bound*.

Se $U^i_{\max} < rl_i \Rightarrow \sum_{j \in \mathcal{B}^+} a^i_j < rl_i$ então o problema é infactível conforme ilustra a Figura A.2.

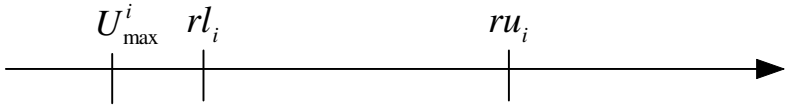

Fig. A.2: Limitante superior inferior ao *lower bound*.

A.2 *Probing* para fixar variáveis em 0 ou 1

Se um problema não falha no teste de infactibilidade tenta-se fixar variáveis pela análise de quatro casos possíveis.

A.2.1 Primeiro caso

Seja $x_k \in \mathcal{B}^+$ na restrição i. Se x_k assumir o valor 1 impondo a restrição $x_k = 1$, teremos novos valores $\bar{L}^i_{\min} = L^i_{\min} + a^i_k$ e $\bar{U}^i_{\max} = U^i_{\max}$ de acordo com o desenvolvimento a seguir.

$$\begin{aligned}
\bar{L}^i_{\min} &= \min \sum_{j \in \mathcal{B}^+ \setminus \{k\}} a^i_j x_j + a^i_k x_k - \sum_{j \in \mathcal{B}^-} a^i_j x_j \\
&= \min \sum_{j \in \mathcal{B}^+ \setminus \{k\}} a^i_j x_j + a^i_k - \sum_{j \in \mathcal{B}^-} a^i_j x_j = +a^i_k - \sum_{j \in \mathcal{B}^-} a^i_j \\
&= L^i_{\min} + a^i_k
\end{aligned}$$

$$\begin{aligned}
\bar{U}^i_{\max} &= \max \sum_{j \in \mathcal{B}^+ \setminus \{k\}} a^i_j x_j + a^i_k x_k - \sum_{j \in \mathcal{B}^-} a^i_j x_j \\
&= \max \sum_{j \in \mathcal{B}^+ \setminus \{k\}} a^i_j x_j + a^i_k - \sum_{j \in \mathcal{B}^-} a^i_j x_j = \sum_{j \in \mathcal{B}^+ \setminus \{k\}} a^i_j + a^i_k = \sum_{j \in \mathcal{B}^+} a^i_j \\
&= U^i_{\max}
\end{aligned}$$

Se $\bar{L}^i_{\min} > ru_i$ então x_k não pode assumir o valor um, ou seja, se $a^i_k + L^i_{\min} > ru_i$ então $x_k \neq 1$.

Se $\bar{U}^i_{\max} < rl_i$ então x_k não pode assumir o valor um, ou seja, se $U^i_{\max} < rl_i$ então $x_k \neq 1$. Mas note que esse teste já foi feito no teste de infactibilidade e assim não precisa ser refeito aqui, pois seria redundante.

A.2.2 Segundo caso

Seja $x_k \in \mathcal{B}^+$ na restrição i. Se x_k assumir o valor 0, impondo a restrição $x_k = 0$, teremos novos valores $\bar{L}^i_{\min} = L^i_{\min}$ e $\bar{U}^i_{\max} = U^i_{\max} - a^i_k$ de acordo com o desenvolvimento a seguir.

$$\begin{aligned}
\bar{L}^i_{\min} &= \min \sum_{j \in \mathcal{B}^+ \setminus \{k\}} a^i_j x_j + a^i_k x_k - \sum_{j \in \mathcal{B}^-} a^i_j x_j \\
&= \min \sum_{j \in \mathcal{B}^+ \setminus \{k\}} a^i_j x_j + 0 \times a^i_k - \sum_{j \in \mathcal{B}^-} a^i_j x_j = - \sum_{j \in \mathcal{B}^-} a^i_j \\
&= L^i_{\min}
\end{aligned}$$

$$\begin{aligned}\bar{U}^i_{\max} &= \max \sum_{j\in\mathcal{B}^+\setminus\{k\}} a^i_j x_j + a^i_k x_k - \sum_{j\in\mathcal{B}^-} a^i_j x_j \\ &= \max \sum_{j\in\mathcal{B}^+\setminus\{k\}} a^i_j x_j + 0 \times a^i_k - \sum_{j\in\mathcal{B}^-} a^i_j x_j \\ &= \sum_{j\in\mathcal{B}^+\setminus\{k\}} a^i_j = \sum_{j\in\mathcal{B}^+} a^i_j - a^i_k \\ &= U^i_{\max} - a^i_k\end{aligned}$$

Se $\bar{U}^i_{\max} < rl_i$ então x_k não pode assumir o valor zero, ou seja, se $U^i_{\max} - a^i_k < rl_i$ então $x_k \neq 0$.

Se $\bar{L}^i_{\min} > ru_i$ então x_k não pode assumir o valor zero, ou seja, se $L^i_{\min} > ru_i$ então $x_k \neq 0$. Mas note que esse teste já foi feito no teste de infactibilidade e assim não precisa ser refeito aqui, pois seria redundante.

A.2.3 Terceiro caso

Seja $x_k \in \mathcal{B}^-$ na restrição i. Se x_k assumir o valor 1, impondo a restrição $x_k = 1$, teremos novos valores $\bar{L}^i_{\min} = L^i_{\min}$ e $\bar{U}^i_{\max} = U^i_{\max} - a^i_k$ de acordo com o desenvolvimento a seguir.

$$\begin{aligned}\bar{L}^i_{\min} &= \min \sum_{j\in\mathcal{B}^+} a^i_j x_j - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j x_j - a^i_k x_k \\ &= \min \sum_{j\in\mathcal{B}^+} a^i_j x_j - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j x_j - a^i_k = - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j - a^i_k \\ &= L^i_{\min}\end{aligned}$$

$$\begin{aligned}\bar{U}^i_{\max} &= \max \sum_{j\in\mathcal{B}^+} a^i_j x_j - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j x_j - a^i_k x_k \\ &= \max \sum_{j\in\mathcal{B}^+} a^i_j x_j - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j x_j - a^i_k = \sum_{j\in\mathcal{B}^+} a^i_j - a^i_k \\ &= U^i_{\max} - a^i_k\end{aligned}$$

Se $\bar{U}^i_{\max} < rl_i$ então x_k não pode assumir o valor um, ou seja, se $U^i_{\max} - a^i_k < rl_i$ então $x_k \neq 1$.

Se $\bar{L}^i_{\min} > ru_i$ então x_k não pode assumir o valor um, ou seja, se $L^i_{\min} > ru_i$ então $x_k \neq 1$. Mas note que esse teste já foi feito no teste de infactibilidade e assim não precisa ser refeito aqui, pois seria redundante.

A.2.4 Quarto caso

Seja $x_k \in \mathcal{B}^-$ na restrição i. Se x_k assumir o valor 0, impondo a restrição $x_k = 0$, teremos novos valores $\bar{L}^i_{\min} = L^i_{\min} + a^i_k$ e $\bar{U}^i_{\max} = U^i_{\max}$ de acordo com o desenvolvimento a seguir.

$$\begin{aligned}\bar{L}^i_{\min} &= \min \sum_{j\in\mathcal{B}^+} a^i_j x_j - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j x_j - a^i_k x_k \\ &= \min \sum_{j\in\mathcal{B}^+} a^i_j x_j - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j x_j - a^i_k \times 0 = - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j \\ &= L^i_{\min} + a^i_k\end{aligned}$$

$$\begin{aligned}\bar{U}^i_{\max} &= \max \sum_{j\in\mathcal{B}^+} a^i_j x_j - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j x_j - a^i_k x_k \\ &= \max \sum_{j\in\mathcal{B}^+} a^i_j x_j - \sum_{j\in\mathcal{B}^-\setminus\{k\}} a^i_j x_j - a^i_k \times 0 = \sum_{j\in\mathcal{B}^+} a^i_j \\ &= U^i_{\max}\end{aligned}$$

Se $\bar{L}^i_{\min} > ru_i$ então x_k não pode assumir o valor zero, ou seja, se $L^i_{\min} + a^i_k > ru_i$ então $x_k \neq 0$.

Se $\bar{U}^i_{\max} < rl_i$ então x_k não pode assumir o valor um, ou seja, se $U^i_{\max} < rl_i$ então $x_k \neq 0$. Mas note que esse teste já foi feito no teste de infactibilidade e assim não precisa ser refeito aqui, pois seria redundante.

A.2.5 Detalhes de implementação

Para as variáveis fixadas, é preciso atualizar ru_i e rl_i como visto anteriormente no teste de infactibilidade. Para as variáveis livres, aplica-se o tratamento a seguir lembrando que α^i_j são os coeficientes na matriz A e $a^i_j = \left|\alpha^i_j\right|$. Calcula-se então os valores $U^i_{\max}$ e $L^i_{\min}$ para cada restrição de acordo com (A.7).

$$U^i_{\max} = \sum_{j\in\mathcal{B}^+} \alpha^i_j = \sum_{j\in\mathcal{B}^+} a^i_j \quad \text{e} \quad L^i_{\min} = \sum_{j\in\mathcal{B}^-} \alpha^i_j = - \sum_{j\in\mathcal{B}^-} a^i_j \tag{A.7}$$

Se $L^i_{\min} > ru_i$ ou se $U^i_{\max} < rl_i$ então o problema é infactível. Caso o PL não seja infactível, pode-se fixar uma variável nas seguintes situações de (A.8).

$$\begin{array}{l} \text{Se } L^i_{\min} + |\alpha^i_k| > ru_i \text{ então fixa } \begin{cases} x_k = 0 \text{ se } \alpha^i_k > 0 \\ x_k = 1 \text{ se } \alpha^i_k < 0 \end{cases} \\ \text{Se } U^i_{\max} - |\alpha^i_k| < rl_i \text{ então fixa } \begin{cases} x_k = 1 \text{ se } \alpha^i_k > 0 \\ x_k = 0 \text{ se } \alpha^i_k < 0 \end{cases} \end{array} \tag{A.8}$$

Se uma variável x_k é fixada em um, ru_i e rl_i precisam ser atualizados para todas as restrições. Além disso independentemente da variável ter sido fixada em zero ou em um, em cada restrição, a medida $U^i_{\max}$ ou $L^i_{\min}$ será atualizada dependendo do sinal do coeficiente α^i_k ser positivo ou negativo.

Note que a análise de fixar uma variável é aplicada em uma restrição. Sendo assim, após fixar uma variável pelas regras, o teste de infactibilidade precisa ser imediatamente aplicado, pois em uma outra restrição, tal fixação em um ou zero pode ter tornado o PL infactível.

Apêndice B

Detalhes da Implementação da Busca Tabu Paramétrica

O primeiro propósito desse apêndice é apresentar detalhes da implementação omitidos nos Capítulos 2 e 3 mas que podem ser relevantes para alguém que se proponha a implementar a busca da maneira mais fiel à utilizada nesta tese. O segundo propósito é mostrar com dados de tempo de execução que o algoritmo apresenta um comportamento assintótico linear com relação ao tamanho da instância.

B.1 Detalhamento do algoritmo de curto prazo

A análise e o projeto da Busca Tabu Paramétrica foram conduzidos de acordo com a orientação a objetos com implementação em linguagem C++. Mas ao contrário de apresentar os algoritmos na forma de classes, por questão de clareza, eles são apresentados numerados por passos. A Tabela B.1 apresenta os parâmetros da busca suscetíveis a calibração. A Tabela B.2 descreve processos internos importantes. Em geral, cada um desses processos foi implementado com uma classe ou com uma hierarquia de classes. As variáveis internas mais importantes estão elencadas na Tabela B.3. A Busca Tabu Paramétrica de curto prazo é apresentada na Tabela B.4. Excepcionalmente, no *Passo* 6 foi incluído o algoritmo de *probing* que faz parte das extensões. Os *Passos* 4 e 5 envolvem os tratamentos de variáveis com infactibilidade de metas e variáveis com infactibilidade inteira. Esses passos são detalhados respectivamente nas Tabelas B.5 e B.6.

Tab. B.1: Parâmetros da BTP.

Parâmetro	Descrição
`Gp_min`	Número mínimo desejado de variáveis em G_p.
`Gs_min`	Número mínimo desejado de variáveis em G_s.
`G_max`	Número máximo permitido de variáveis em G_p ou G_s quando o critério adaptativo de controle de cardinalidades é usado.
`maxPotentialGRVars`	Número máximo aceito de variáveis com infactibilidade potencial de metas.
`Do_min`	Número mínimo de variáveis aceitas para compor o conjunto D_0.
`fd`	Fração da cardinalidade do conjunto D indicando as variáveis com maiores valores preferenciais de escolha a serem selecionadas para compor o conjunto D_0.
`fp`	Fração da cardinalidade do conjunto G indicando as variáveis com maiores valores de resistência a metas a serem selecionadas para compor o conjunto G_p.
`fs`	Fração da cardinalidade do conjunto G. Para efeito de seleção, considera apenas as variáveis em $G - G_p$ com maiores valores de resistência a metas para compor o conjunto G_s.
`maxIter`	Número máximo de iterações para o critério de parada baseado em número máximo de iterações transcorridas.
`timeLimit`	Tempo máximo em segundos para o critério de parada baseado em tempo máximo transcorrido.
`minTenure`	Número mínimo de duração tabu aplicável a uma condição de meta.
`maxTenure`	Número máximo de duração tabu aplicável a uma condição de meta.
Δ`-Tenure`	Quando uma variável torna-se tabu UP ou DN, incrementa-se o número de iterações tabu para a condição de meta no sentido oposto DN ou UP, respectivamente em Δ`-Tenure`, sem nunca ultrapassar `maxTenure`.
`maxClearTabuIter`	Após `maxClearTabuIter` iterações, o gerenciador de durações tabu é zerado de forma que todas as durações tabu voltam ao valor inicial `minTenure`.
`epsilon`	Quando a restrição de função objetivo é atualizada após a descoberta de uma nova solução incumbente, exclui essa solução da região factível do PL'.

Tab. B.2: Processos importantes da BTP.

Processo	Descrição
`tabu_tenure_generator`	É um processo que controla a duração tabu estabelecida para variáveis com infactibilidade de metas. Quando uma solução incumbente é encontrada ou após um número máximo de iterações transcorridas, os valores mínimos de duração tabu são restabelecidos. Os detalhes da implementação estão na Subseção 2.3.11.
`Gp_Gs_Do_manager`	Controla a cardinalidade dos conjuntos G_p, G_s e D_0. A abordagem estática que extrai frações de G e D (f_p, f_s e f_d) ou a abordagem adaptativa é aplicada de acordo com a Subseção 2.3.13.

Continua na próxima página.

Tab. B.2 – Continuação da página anterior.

`stop_criterion`	Um critério de parada. Pode ser por número máximo de iterações, ou tempo máximo transcorrido.
`overt_goal_resistance_alg`	Computa a resistência direta a metas, pode ser por vizinho inteiro mais próximo (DM), *Reliability Branching* (RB) ou *Active Constraint Variable Ordering* (ACVO) de acordo com a Subseção 2.3.12.
`pot_goal_resistance_alg`	Computa resistência potencial a metas e foi conduzida pela análise de custo reduzido discutida na Subseção 2.3.6.
`integer_penalty_alg`	Computa infactibilidade inteira, pode ser por DM, RB ou ACVO de acordo com a Subseção 2.3.12.
`probing`	Algoritmo detalhado no Apêndice A baseado na fixação de variáveis em 1 ou 0. Como a busca é paramétrica, ao contrário de fixar variáveis, metas paramétricas são estabelecidas. • Numa mesma iteração, pode haver metas detectadas por `probing` e metas estabelecidas pela BTP. Elas são unidas em um único conjunto, prevalecendo a meta criada pelo *probing* quando há conflito. • Se o `probing` sugere uma transição que entra em conflito com uma meta em sentido oposto estabelecida em iterações anteriores da busca, o `probing` é desativado completamente nessa iteração da BTP.

Tab. B.3: Variáveis importantes.

Variável	Descrição
`binaryVarNumber`	Número de variáveis binárias existentes em uma instância.
`binaryVarIndex[]`	Vetor com os índices das colunas ocupadas pelas variáveis binárias na matriz do PIM.
`binaryVarValue[]`	Valores assumidos pelas variáveis binárias em uma instância do PL', ou seja, x''.
x_0^*	Valor incumbente de função objetivo do PIM.
`numberOfOvertGRVars`	Número de variáveis com infactibilidade de metas identificadas para um PL'.
`numberOfPotentialGRVars`	Número de variáveis com infactibilidade potencial de metas identificadas para um PL'. Só são identificadas quando `numberOfOvertGRVars` > 0. $\|G\|$ =`numberOfOvertGRVars`+`numberOfPotentialGRVars`.
`numberOfIntInfeasibleVars`	Número de variáveis irrestritamente fracionárias. Só são identificadas quando $\|G\| = 0$. $\|D\| =$ `numberOfIntInfeasibleVars`.
`goalValue[]`	Vetor que mantém os valores de metas designados às variáveis inteiras. É o vetor x'.
`newGoalValue`	Conjunto auxiliar no qual tanto as transições sob infactibilidade de metas quanto as transições sob infactibilidade inteira são inseridas para a criação do PL' da próxima iteração.

Continua na próxima página.

Tab. B.3 – Continuação da página anterior.

Variável	Descrição
`aspirationUP[]`	Vetor que mantém os valores de aspiração sob resistência para transições UP. Para cada variável, um par de valores é mantido. O valor numérico da resistência e um booleano que indica se tal valor foi originado por uma resistência direta ou potencial.
`aspirationDN[]`	Vetor que mantém os valores de aspiração sob resistência para transições DN. Para cada variável, um par de valores é mantido. O valor numérico da resistência e um booleano que indica se tal valor foi originado por uma resistência direta ou potencial.
`goalResistanceUP[]`	Vetor que mantém as medidas de resistência UP, quando ocorre uma V-UP. Para cada variável, um par de valores é mantido. O valor numérico da resistência e um booleano que indica se tal valor foi originado por uma resistência direta ou potencial.
`goalResistanceDN[]`	Vetor que mantém as medidas de resistência DN, quando ocorre uma V-DN. Para cada variável, um par de valores é mantido. O valor numérico da resistência e um booleano que indica se tal valor foi originado por uma resistência direta ou potencial.
`integerPenaltyUP[]`	Vetor que mantém a medida de penalidade inteira para variáveis irrestritamente fracionárias caso uma transição T-UP seja aplicada.
`integerPenaltyDN[]`	Vetor que mantém a medida de penalidade inteira para variáveis irrestritamente fracionárias caso uma transição T-DN seja aplicada.
`iter`	Iteração corrente.
`bigM_pattern[]`	Uma seqüência de valores de M grande é mantida em um vetor de forma que esses valores apresentem um comportamento de queda monotônica à medida que os índices aumentam no vetor. Adotou-se um decaimento exponencial como padrão. Quando as metas paramétricas são estabelecidas, recebem o maior valor de M grande, `bigM_pattern[0]` , na iteração seguinte, tais variáveis terão seus M grandes atualizados para `bigM_pattern[1]` e assim por diante. Assim, à medida que as iterações do método progridem, as variáveis parametrizadas têm valores de M grande diminuídos, colocando assim maior ênfase nas metas mais recentemente estabelecidas. Ver a Subseção 2.3.2.
`last_transition[]`	Para cada variável 0–1 guarda em que iteração a última transição UP ou DN foi estabelecida. Transição FREE não atualiza esse vetor.
`solution[]`	Solução do PL', ou seja, (x'', y'').

Tab. B.4: Algoritmo da Busca Tabu Paramétrica.

Passo 0

1. Identifique as variáveis binárias e inicialize as variáveis de controle.
   - `binaryVarNumber`
   - `binaryVarIndex[1,...,binaryVarNumber]`
   - $x_0^* \leftarrow \infty$
   - `iter` $\leftarrow$ `0`
   - `bigM_pattern[]`$\leftarrow$ `{seqüência de decaimento exponencial}`
2. Inicialize as estruturas de dados para $j \in \{1,\ldots,$`binaryVarNumber`$\}$
   - `aspirationUP[j]` $\leftarrow$ `(-`∞`, Potential)`
   - `aspirationDN[j]` $\leftarrow$ `(-`∞`, Potential)`
   - `goalValue[j]` $\leftarrow$ `FREE`
   - `goalResistanceUP[j]` $\leftarrow$ `(-`∞`,Potential)`
   - `goalResistanceDN[j]` $\leftarrow$ `(-`∞`,Potential)`
   - `integerPenaltyUP[j]` $\leftarrow$ `-`∞
   - `integerPenaltyDN[j]` $\leftarrow$ `-`∞
   - `tabuFimUP[j]` $\leftarrow$ `-1`
   - `tabuFimDN[j]` $\leftarrow$ `-1`
3. Resolva a relaxação linear PL do PIM e obtenha x''.
   - para $j \in \{1,\ldots,$`binaryVarNumber`$\}$
     - `binaryVarValue[j]` $\leftarrow$ `solution[binaryVarIndex[j]]`

Passo 1

- `newGoalValue` $\leftarrow$ $\emptyset$
- Se o critério de parada, `stop_criterion`, for satisfeito, vá para o *Passo* 8.

Passo 2 Se o PL' for infactível

- A última solução inteira encontrada é ótima, desde que `epsilon` seja suficientemente pequeno.
- Vá para o *Passo* 8.

Continua na próxima página.

Tab. B.4 – Continuação da página anterior.

Passo 3	Se a solução do PL' for inteira factível • Atualize o valor da solução incumbente, $x_0^* \leftarrow x_0$ • Atualize a restrição de função objetivo, $cx + dy \leq x_0^*$−`epsilon`. • Reinicialize o gerador de durações tabu, `tabu_tenure_generator`. • Reinicialize os vetores de aspiração. Para $j \in \{1, \ldots,$`binaryVarNumber`$\}$ – `aspirationUP[j]` $\leftarrow$ `(-`∞`, Potential)` – `aspirationDN[j]` $\leftarrow$ `(-`∞`, Potential)` • Reotimize o PL' com a abordagem de duas fases. • Vá para o *Passo* 1.
Passo 4	Encontre as infactibilidades de metas com os algoritmos `overt_goal_resistance_alg` e `pot_goal_resistance_alg`. Se G não for vazio. • Crie G_p e G_s usando o `Gp_Gs_Do_manager` disponível. • Crie as transições sob infactibilidade de metas a partir dos conjuntos G_p e G_s, inserindo as transições T-UP, T-DN e T-FREE no conjunto `newGoalValue`. • Vá para o *Passo* 6.
Passo 5	Encontre as variáveis irrestritamente fracionárias, calculando as penalidades inteiras com o algoritmo `integer_penalty_alg`. Note que $G = \emptyset$ ao alcançar esse passo. • Crie D_0 usando o `Gp_Gs_Do_manager` disponível. • Crie as transições sob infactibilidade inteira a partir do conjunto D_0, inserindo as transições T-UP e T-DN no conjunto `newGoalValue`.
Passo 6	Aplique `Probing`. • Se algum conflito com uma meta estabelecida em iterações passadas for identificado, anule todo o processo de *probing* nessa iteração. • As metas *probing* são inseridas em `newGoalValue` mesmo se entrarem em conflito com outras metas estabelecidas nos *Passos* 4 ou 5 dessa iteração.

Continua na próxima página.

Tab. B.4 – Continuação da página anterior.

Passo 7	• Crie as novas transições registradas no conjunto `newGoalValue`, produzindo assim um novo PL'. • Para cada variável em `newGoalValue`, com transição UP ou DN, atualizar a posição correspondente no vetor `last_transition[]`. • Reotimize o novo PL' com a abordagem de duas fases. • Se mais de `resetAfterIter` iterações transcorreram desde a última reinicialização de durações tabu: – Reinicialize o `tabu_tenure_generator`. – Reinicialize os vetores de aspiração. Para $j \in \{1,\ldots,$`binaryVarNumber`$\}$ ∗ `aspirationUP[j]` ← `(-`∞`, Potential)` ∗ `aspirationDN[j]` ← `(-`∞`, Potential)` • `iter` ← `iter + 1.` • Vá para o *Passo* 1.
Passo 8	• Apresente x_0^* como o melhor valor encontrado de função objetivo. • Fim

Tab. B.5: Detalhamento do *Passo* 4 – Cômputo da resistência a metas.

Passo 0	• `current_G_size` ← `0` • `numberOfOvertGRVars` ← `0` • `numberOfPotentialGRVars` ← `0` • `atLeastOneVarIn_G` ← **`falso`** • `lessTabuOvertGRVarIndex` ← `-1` • `lessTabuPotentialGRVarIndex` ← `-1` • `smallestResidualTabuTenure` ← ∞ • `j` ← `1`
Passo 1	• `goalResistanceUP[j]` ← `(-`∞`,Potential)` • `goalResistanceDN[j]` ← `(-`∞`,Potential)` • Se `(goalValue[j]= FREE) ou (binaryVarValue[j] = goalValue[j])` – Vá para o *Passo* 4.

Continua na próxima página.

Tab. B.5 – Continuação da página anterior.

Passo 2 Variável não-tabu: Se $\texttt{max\{FimTabu}_j\texttt{(DN),FimTabu}_j\texttt{(UP)\}} \leq \texttt{iter}$

- `atLeastOneVarIn_G` ← **verdadeiro**
- `current_G_size` ← `current_G_size + 1`
- Compute a resistência direta à meta UP ou DN para a variável `binaryVarIndex[j]` usando o algoritmo `overt_goal_resistance_alg`. Mantenha o valor calculado em `goalResistanceUP[j]` ou `goalResistanceDN[j]`.
- `numberOfOvertGRVars` ← `numberOfOvertGRVars + 1`
- Vá para o *Passo* 4.

Passo 3 Variável tabu

- Compute a resistência direta à meta UP ou DN para a variável `binaryVarIndex[j]` usando o algoritmo `overt_goal_resistance_alg`. Mantenha o valor calculado em `goalResistanceUP[j]` ou `goalResistanceDN[j]`.
- Verificar se o critério de aspiração por resistência é satisfeito dependendo da variável apresentar V-UP ou V-DN:
  - Se `goalResistanceUP[j]` > `aspirationDN[j]`
    - `current_G_size` ← `current_G_size + 1`
    - `atLeastOneVarIn_G` ← **verdadeiro**
    - `numberOfOvertGRVars` ← `numberOfOvertGRVars + 1`
  - Se `goalResistanceDN[j]` > `aspirationUP[j]`
    - `current_G_size` ← `current_G_size + 1`
    - `atLeastOneVarIn_G` ← **verdadeiro**
    - `numberOfOvertGRVars` ← `numberOfOvertGRVars + 1`
  - Se `atLeastOneVarIn_G` = **falso** e $\texttt{max\{FimTabu}_j\texttt{(DN),FimTabu}_j\texttt{(UP)\}} - \texttt{iter} < \texttt{smallestResidualTabuTenure}$
    - `smallestResidualTabuTenure` ← $\texttt{max\{FimTabu}_j\texttt{(DN),FimTabu}_j\texttt{(UP)\}} - \texttt{iter}$
    - `lessTabuOvertGRVarIndex` ← `j`

Passo 4

- `j` ← `j + 1`
- Se `j` ≤ `binaryVarNumber`
  - Vá para o *Passo* 1.

Continua na próxima página.

Tab. B.5 – Continuação da página anterior.

Passo 5	Tratamento de variáveis potencialmente infactíveis com relação a metas. • Se `atLeastOneVarIn_G = falso e lessTabuOvertGRVarIndex = -1` – Vá para o *Passo* 14.
Passo 6	• `j ← 1`
Passo 7	• Se `(goalValue[j] = FREE) ou (binaryVarValue[j] ≠ goalValue[j])` – Vá para o *Passo* 10.
Passo 8	Variável não-tabu. • Se `max{FimTabu`$_j$`(DN),FimTabu`$_j$`(UP)} ≤ iter` – `atLeastOneVarIn_G ← verdadeiro` – `current_G_size ← current_G_size + 1` – Compute a resistência potencial à meta UP ou DN para a variável `binaryVarIndex[j]` usando o algoritmo `pot_goal_resistance_alg.` Mantenha o valor calculado em `goalResistanceUP[j]` ou `goalResistanceDN[j].` – `numberOfPotentialGRVars ← numberOfPotentialGRVars + 1` – Vá para o *Passo* 10.

Continua na próxima página.

Tab. B.5 – Continuação da página anterior.

Passo 9 Variável tabu.

- Compute a resistência potencial à meta UP ou DN para a variável `binaryVarIndex[j]` usando o algoritmo `pot_goal_resistance_alg`. Mantenha o valor calculado em `goalResistanceUP[j]` ou `goalResistanceDN[j]`.
- Verificar se o critério de aspiração por resistência é satisfeito dependendo da variável apresentar V-UP ou V-DN:
  - Se `goalResistanceUP[j]` > `aspirationDN[j]`
    - `current_G_size` ← `current_G_size + 1`
    - `atLeastOneVarIn_G` ← **`verdadeiro`**
    - `numberOfPotentialGRVars` ← `numberOfPotentialGRVars + 1`
  - Se `goalResistanceDN[j]` > `aspirationUP[j]`
    - `current_G_size` ← `current_G_size + 1`
    - `atLeastOneVarIn_G` ← **`verdadeiro`**
    - `numberOfPotentialGRVars` ← `numberOfPotentialGRVars + 1`
  - Se `atLeastOneVarIn_G` = **`falso`** e $\max\{FimTabu_j(DN), FimTabu_j(UP)\}$ − `iter` < `smallestResidualTabuTenure`
    - `smallestResidualTabuTenure` ← $\max\{FimTabu_j(DN), FimTabu_j(UP)\}$ − `iter`
    - `lessTabuPotentialGRVarIndex` ← `j`

Passo 10

- `j` ← `j + 1`
- Se `j` ≤ `binaryVarNumber`
  - Vá para o *Passo* 7.

Passo 11 Aspiração por *default*.

- Se `atLeastOneVarIn_G` = **`falso`**
  - Se `lessTabuPotentialGRVarIndex` ≥ `0`
    - `current_G_size` ← `1`
    - `numberOfPotentialGRVars` ← `1`
  - Senão se `lessTabuOvertGRVarIndex` ≥ `0`
    - `current_G_size` ← `1`
    - `numberOfOvertGRVars` ← `1`

Continua na próxima página.

Tab. B.5 – Continuação da página anterior.

Passo 12	Ajuste do tamanho de G para atender objetivos conflitantes: `Gp_min, Gs_min` e `maxPotentialGRVars`. • Se `numberOfPotentialGRVars > maxPotentialGRVars` – `delta ← numberOfPotentialGRVars - maxPotentialGRVars` – `current_G_size ← min(current_G_size,max(Gp_min + Gs_min,current_G_size - delta)` – `numberOfPotentialGRVars ← current_G_size - numberOfOvertGRVars`
Passo 13	• Ordene em uma lista as variáveis em G em ordem não-crescente segundo os seus valores de resistência a metas. O `Gp_Gs_Do_manager` instalado insere as primeiras da lista em G_p e as subseqüentes em G_s de acordo com a abordagem estática ou adaptativa. Ao estabelecer em G_p uma transição T-UP ou T-DN, respectivamente, a restrição tabu $\texttt{FimTabu}_j\texttt{(DN)}$ ou $\texttt{FimTabu}_j\texttt{(UP)}$ é definida. Em G_s há apenas transições T-FREE, para as quais não são criadas condições tabu. A cada iteração, os M grandes das variáveis com metas são atualizados com `bigM_pattern[iter-ultimaTransição[j]]`. Transições que se mantêm por mais iterações do que as comportadas pelo vetor `bigM_pattern,` recebem um M grande *default* igual a 1.
Passo 14	• Fim.

Tab. B.6: Detalhamento do *Passo* 5 – Cômputo da infactibilidade inteira.

Passo 0	• `numberOfIntInfeasibleVars ← 0` • `j ← 1`
Passo 1	• `integerPenaltyUP[j] ← -∞` • `integerPenaltyDN[j] ← -∞` • Se `binaryVarValue[j] =` $\lfloor$`binaryVarValue[j]`$\rfloor$ – Vá para o *Passo* 3.
Passo 2	• `numberOfIntInfeasibleVars ← numberOfIntInfeasibleVars + 1` • Compute as penalidades inteiras IP-UP e IP-DN para a variável `binaryVarIndex[j]`. • A partir das penalidades IP-UP e IP-DN, compute CP_j, o valor de escolha preferencial.

Continua na próxima página.

Tab. B.6 – Continuação da página anterior.

Passo 3	• `j ← j + 1` • Se `j` $\leq$ `binaryVarNumber` – Vá para o *Passo* 1.
Passo 4	• Ordene em uma lista as variáveis em D em ordem não-crescente segundo os seus valores de escolha preferencial. O `Gp_Gs_Do_manager` instalado insere as primeiras da lista em D_0 de acordo com a regra que escolhe a cardinalidade do conjuntos D_0. Ao estabelecer T-UP ou T-DN, não são criadas condições tabu. A cada iteração, os M grandes das variáveis com metas são atualizados com `bigM_pattern[iter-ultimaTransição[j]]`. Transições que se mantêm por mais iterações do que o tamanho do vetor `padroes_bigM`, recebem um M grande *default* igual a 1.
Passo 5	• Fim.

B.2 Resumo dos parâmetros da busca tabu paramétrica

Na Tabela B.7 são listados todos os parâmetros de curto prazo, extensões e longo prazo da busca tabu paramétrica. Os parâmetros das estratégias que fazem parte da implementação de referência estão destacados.

Tab. B.7: Parâmetros da busca tabu paramétrica.

Restrição de função objetivo	
Parâmetro	valor
ε	1
Decaimento exponencial de M_j	
Parâmetro	valor
r	0,2
NI	64
Variáveis potencialmente infactíveis com relação a metas	
Parâmetro	valor
T_0	1
Medida de preferência de escolha 1	
Parâmetro	valor
w	0,01
Medida de preferência de escolha 2	
Parâmetro	valor
μ	5/6

Continua na próxima página.

Tab. B.7 – Continuação da página anterior.

Memória tabu	
Parâmetro	valor
k_1	0,05
k_2	0,01
k_3	0,1
MaxLimpaTabuIter	2000
Reliability branching	
Parâmetro	valor
η_{rel}	8
λ	8
γ	10
Cardinalidades estáticas dos conjuntos G_p, G_s e D_0	
Parâmetro	valor
f_p	0,05
f_s	0,05
f_d	0,05
$d_{\min}$	1
$g_{p_{\min}}$	1
$g_{s_{\min}}$	1
Controle adaptativo de G_p, G_s e D_0	
Parâmetro	valor
k_4	0,25
$g_{p_{\min}}$	1
$g_{s_{\min}}$	0
$f_{d_{\max}}$	0,3
δ_1	0,5
δ_2	1,05
f_s	0,05
$g_{\max}$	3
Atualização na restrição de função objetivo	
Parâmetro	valor
Δss	50
MaxRelax	0,03
Δrm	500
Uso de cortes	
Parâmetro	valor
MaxCortes	40
MinDuracaoCortes	40

Continua na próxima página.

Tab. B.7 – Continuação da página anterior.

B&C para resolver infactibilidade de metas	
Parâmetro	valor
MultRR	3
FracPermitida	0,2
FracPorIter	0,1
TempoRConf	10
Conjunto de referência	
Parâmetro	valor
b	29
ω	1,3
ξ	0,7
Matriz de freqüência	
Parâmetro	valor
α_cc	2,46
α_int	0,15
α_div	0,15
Geração de centros por RK_1 e RK_2	
Parâmetro	valor
k	3
f	0,3
V_{min}	2
V_{max}	8
Variáveis consistentes e fortemente determinadas	
Parâmetro	valor
maxRk	3
nivelMax	3
fracaoLimite	0,95
maxAnalisar	0,04
maxFix	0,015
γ_2	50

B.3 Implementação otimista do PL de duas fases

Para implementar o PL de duas fases proposto na Subseção 2.3.2, dois modelos são mantidos em memória. Um parametrizado para resolver a fase I e um segundo para a fase II no qual apenas os limitantes das variáveis são atualizados conforme o resultado da fase I. Embora o uso de dois modelos consuma o dobro de memória, o ganho em tempo computacional se justifica. Note que as

bases simplex correntes e que são utilizadas nas reotimizações ficam automaticamente mantidas em cada modelo.

No exemplo da Seção 2.4, durante as iterações de 1 a 5, o PL' sempre se manteve infactível inteiro. Como as metas estabelecidas na fase I de cada uma dessas iterações foram satisfeitas, a implicação imediata foi de apenas fixá-las nos valores desejados para a execução da fase II. Isso sugere a seguinte implementação otimista[1] do PL de duas fases. Ao contrário de sempre resolver a fase I, cada iteração inicia diretamente na fase II fixando as variáveis nos valores de suas metas. É uma visão otimista de que tais metas podem ser simultaneamente atendidas. Eventualmente quando o conjunto de metas não puder ser atendido, o PL' torna-se infactível nas metas, e surge a infactibilidade explícita do PL' corrente da fase II. Nessa situação, paga-se o preço pelo otimismo. Será preciso executar a fase I, fixar as variáveis com metas atendidas, deixar canalizadas entre 0 e 1 aquelas que não puderam atender as suas metas e resolver novamente para a mesma iteração a fase II, ou seja, nesse caso três PLs foram resolvidos em uma única iteração. O diagrama de estados que representa esse modelo está ilustrado na Figura B.1.

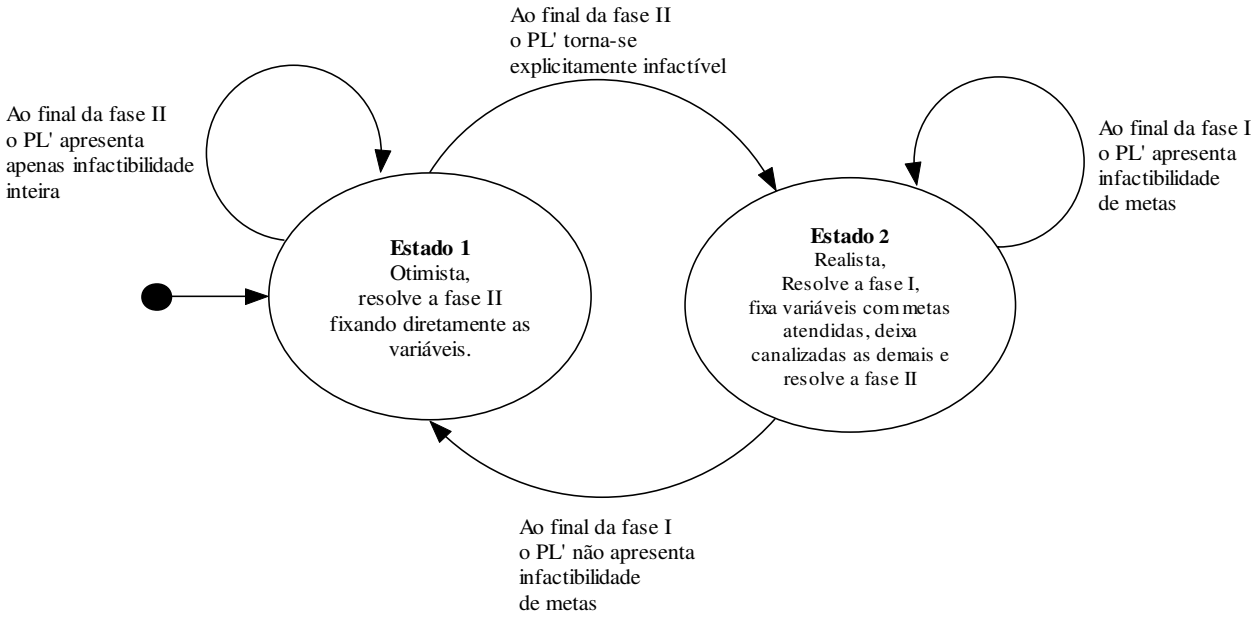

Fig. B.1: Diagrama de estados para implementação otimista do PL de duas fases.

[1]Também poderia ser chamada de fase I preguiçosa por tentar poupar tempo ao não executá-la.

B.4 Comportamento assintótico da implementação de curto prazo

As avaliações que seguem foram conduzidas com a implementação de referência de curto prazo, BTP-ACVO, com tempo total de 3600s. Ficaram ativados o decaimento exponencial de M_j grande, a aspiração por resistência e a abordagem adaptativa de controle de cardinalidade dos conjuntos G_p, G_s e D_0. As instâncias consideradas foram as do tipo GAP utilizadas no Capítulo 5, excluído o grupo A cujas instâncias por serem fáceis têm as execuções encerradas prematuramente com PL' infactível para quatro das seis instâncias devido à atualização na restrição de função objetivo.

A resolução do PL' de duas fases consumiu em média 56% do tempo total. Em média, o cálculo da infactibilidade inteira ocupou 4% do tempo e o cálculo da resistência a metas absorveu 31% do tempo. Sendo assim, 91% do tempo é gasto com esssas três atividades e o restante é dispendido em outras pequenas tarefas como atualizar os pesos da função objetivo, por exemplo.

O uso da abordagem otimista para a resolução do PL' de duas fases merece algumas observações. Ativar a abordagem otimista permite realizar em média, apenas 1,1% a mais de iterações. Esse ganho pequeno é explicado pelo fato da busca passar pouco tempo na infactibilidade inteira, conforme observado no parágrafo anterior. Mas no início da busca, a fase II preguiçosa é particularmente vantajosa devido ao predomínio da infactibilidade inteira, uma vez que as metas estabelecidas ainda não são suficientes para entrarem em conflito uma com as outras. Quando o uso da fase II preguiçosa é desabilitado o tempo para encontrar a primeira solução inteira-factível aumenta em média 9% e ocorrem falhas para as instâncias c801600 e d60900. Desconsiderando essas duas, para as demais 46 instâncias, as somas dos tempos para encontrar a primeira solução foram de 9049,96s e 9621,65s com fase II preguiçosa habilitada e desabilitada respectivamente.

Com relação à complexidade dos algoritmos, Vanderbei (1997) apresenta uma implementação do método dual simplex que consome em média $(m + n)/2$ iterações simplex. Os processos da Busca Tabu Paramétrica executam a cada iteração um total de operações que é proporcional ao número de variáveis binárias. Além disso, há algumas situações envolvendo ordenação de vetores. Como a ordenação de um vetor com m elementos tem complexidade $\mathcal{O}(m \log m)$, é importante apontar os locais que usam a ordenação. As resoluções da infactibilidade de metas e infactibilidade inteira envolvem ordenar um vetor com m itens no pior caso da não-degenerescência. A resolução da infactibilidade potencial de metas envolve ordenar todas as variáveis com metas atendidas e caso todas as variáveis tenham metas estabelecidas, no pior caso haveria $n - 1$ elementos.

Como a análise em curso considera instâncias do tipo GAP, o número n de colunas é bastante superior ao número m de restrições. Para p agentes e q tarefas, com $p \ll q$, há $n = p \times q$ colunas e $m = p + q + 1$ restrições. Para as maiores instâncias de GAP utilizadas, $n = 80 \times 1600$ e $m = 80 + 1600 + 1$, levando a uma relação $\frac{n}{m} = 76,1451517$. Com essa observação, assumiu-se a hipótese de que para esse universo de instâncias o tempo gasto com ordenação não teria influência

suficiente para perturbar a característica linear das demais componentes da BTP. Para um total de 2468 iterações, coletou-se então os tempos de execução para as 51 instâncias de GAP, excluído o grupo A. Considerou-se o tamanho de cada instância como sendo $m + n = p \times q + p + q + 1$ e com uso de um gráfico de dispersão, o tamanho de cada instância foi registrado em relação ao tempo consumido. O gráfico encontra-se na Figura B.2 e contempla a reta $y = 0{,}0278x + 59{,}117$ derivada por regressão linear com o método dos mínimos quadrados. Note que o valor 0,9902 de R-quadrado está muito próximo de um mostrando um forte relacionamento linear entre as amostras avaliadas.

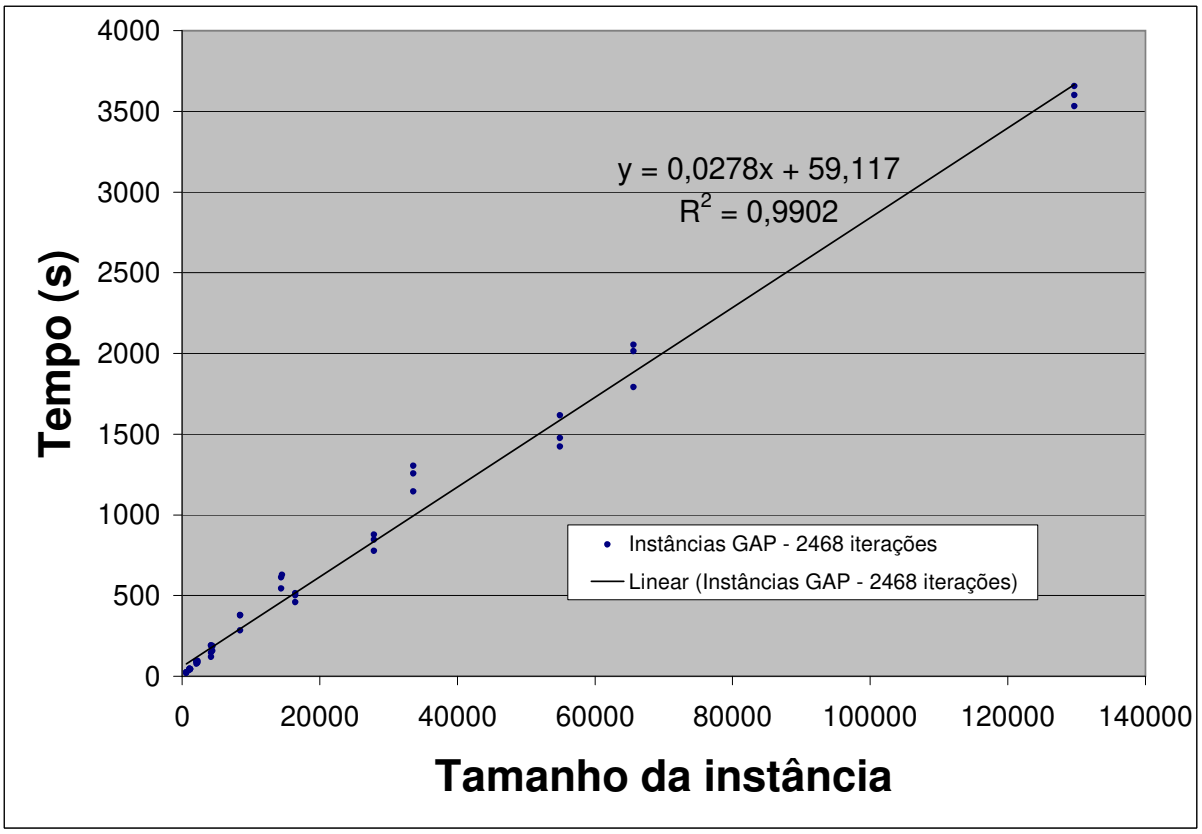

Fig. B.2: Comportamento assintótico da implementação da Busca Tabu Paramétrica.

B.5 Algumas notas de implementação sobre a construção da matriz de freqüência

Se a matriz de freqüência discutida na Subseção 3.2.2 for criada e mantida em memória, os cálculos de $InScore_h(\gamma)$ ficam mais rápidos, e a matriz só precisa ser parcialmente atualizada quando algum elemento em R é substituído. A linha j e a coluna j da matriz de freqüência precisam ser

atualizadas se a variável x_j apresentar valores distintos na solução entrando e na solução saindo do conjunto R. A matriz também apresenta uma simetria de forma que pode ser armazenada como uma matriz triangular superior. Como b não é muito grande, pode-se usar bytes para armazenar cada elemento da matriz, e para n variáveis binárias serão consumidos $(1 + 2 + \ldots + (n-1)) \times 4$ bytes, ou seja, $2n\,(n-1)$ bytes. Se por exemplo, $n = 38730$, serão consumidos 2.999.948.340 bytes, praticamente 3Gbytes, inviabilizando o tratamento para algumas instâncias utilizadas. No Apêndice C, na Tabela C.1 há uma coluna com o número de variáveis binárias de cada instância.

Apêndice C

Informações Sobre as Instâncias

Algumas informações sobre as instâncias utilizadas no Capítulo 4 são apresentadas nesse apêndice. A Tabela C.1 traz os dados sobre as instâncias utilizadas por Achterberg e Berthold (2007) e a Tabela C.2 traz os dados das instâncias usadas por Ghosh (2007). A coluna **Instância** traz o nome das instâncias, a coluna **Melhor valor**, o valor de função objetivo utilizado como referência e pode variar para a mesma instância conforme o autor. Em seguida são apresentadas as colunas referentes ao número de variáveis binárias, número de variáveis contínuas, número de linhas, número de colunas e a quantidade de elementos não-zero na matriz. Na Tabela C.1, a coluna **Status** pode ser *green* (G), *yellow* (Y) ou *red* (R). As instâncias **G** são resolvidas pelo *branch-and-cut* do Cplex em menos de uma hora, as instâncias **Y** são resolvidas na otimalidade, mas com tempo superior a uma hora e as instâncias **R** ainda não tiveram a otimalidade comprovada para a melhor solução encontrada. As instâncias de Mittelmann (2003) não recebem esse tipo de classificação e aparecem com o símbolo "–". Não há necessidade de apresentar os dados sobre as instâncias GAP, MKP e MDMKP porque a nomenclatura utilizada para descrevê-las já contempla esses dados.

Tab. C.1: Dados sobre as instâncias da Miplib utilizadas.

Instância	Melhor valor	Variáveis binárias	Variáveis contínuas	Linhas	Colunas	Não-zeros	Status
10teams	924	1800	225	230	2025	12150	G
a1c1s1	11503,4	192	3456	3312	3648	10178	Y
aflow30a	1158	421	421	479	842	2091	G
aflow40b	1168	1364	1364	1442	2728	6783	Y
air04	56137	8904		823	8904	72965	G
air05	26374	7195		426	7195	52121	G
cap6000	-2451377	6000		2176	6000	48243	G
dano3mip	714,125	552	13321	3202	13873	79655	R
danoint	65,67	56	465	664	521	3232	Y
disctom	-5000	10000		399	10000	30000	G
ds	412,5025	67732		656	67732	1024059	R
fast0507	174	63009		507	63009	409349	G
fiber	405935,18	1254	44	363	1298	2944	G
fixnet6	3983	378	500	478	878	1756	G
glass4	1,20E+09	302	20	396	322	1815	Y
harp2	-73899798	2993		112	2993	5840	G
liu	1496	1089	67	2178	1156	10626	R
markshare1	1	50	12	6	62	312	Y

Continua na próxima página.

Tab. C.1 – Continuação da página anterior.

Instância	Melhor valor	Variáveis binárias	Variáveis contínuas	Linhas	Colunas	Não-zeros	Status
markshare2	1	60	14	7	74	434	Y
mas74	11801,186	150	1	13	151	1706	G
mas76	40005,054	150	1	12	151	1640	G
misc07	2810	259	1	212	260	8619	G
mkc	-563,846	5323	2	3411	5325	17038	Y
mod011	-54558535	96	10862	4480	10958	22254	G
modglob	20740508	98	324	291	422	968	G
momentum1	109143	2349	2825	42680	5174	103198	Y
net12	214	1603	12512	14021	14115	80384	Y
nsrand-ipx	51200	6620	1	735	6621	223261	Y
nw04	16862	87482		36	87482	636666	G
opt1217	-16	768	1	64	769	1542	G
p2756	3124	2756		755	2756	8937	G
pk1	11	55	31	45	86	915	G
pp08a	7350	64	176	136	240	480	G
pp08aCUTS	7350	64	176	246	240	839	G
protfold	-31	1835		2112	1835	23491	Y
qiu	-132,87314	48	792	1192	840	3432	G
rd-rplusc-21	165395	457	165	125899	622	852384	Y
set1ch	54537,75	240	472	492	712	1412	G
seymour	423	1372		4984	1372	33549	Y
sp97ar	6,75E+08	14101		1761	14101	290968	R
stp3d		204880		159488	204880	662128	R
swath	467,407	6724	81	884	6805	34965	Y
t1717	195779	73885		551	73885	325689	R
tr12-30	130596	360	720	750	1080	2508	Y
vpm2	13,75	168	210	234	378	917	G
l152lav	4722	1989		97	1989	9922	–
stein45	30	45		331	45	1034	–
ran8x32	5247	256	256	296	512	1024	–
ran10x26	4270	260	260	296	520	1040	–
ran12x21	3664	252	252	285	504	1008	–
ran13x13	3252	169	169	195	338	676	–
binkar10_1	6742,2	170	2128	1026	2298	4496	–
irp	12159,493	20315	0	39	20315	98254	–
mas284	91405,724	150	0	68	150	9631	–
prod1	-56	149	101	208	250	5350	–
bc1	3,3383625	252	1499	1913	1751	276842	–
bienst1	46,75	28	477	576	505	2184	–
bienst2	54,6	35	470	576	505	2184	–
dano3_3	576,34463	69	13804	3202	13873	79655	–
dano3_4	576,43522	92	13781	3202	13873	79655	–
dano3_5	576,92492	115	13758	3202	13873	79655	–
mkc1	-607,15	3087	2238	3411	5325	17038	–
neos1	19	2112		5020	2112	21312	–
neos2	454,8647	1040	1061	1103	2101	7326	–
neos3	368,84275	1360	1387	1442	2747	9580	–
neos4	-4,86E+10	17172	5712	38577	22884	99930	–
neos5	15	17136	3988	36702	21124	93681	–
neos6	83	8340	446	1036	8786	251946	–
seymour1	410,7637	451	921	4944	1372	33549	–
swath1	379,071	2306	4499	884	6805	34965	–
swath2	385,19969	2406	4399	884	6805	34965	–
acc-0	0	1620		1737	1620	7290	–
acc-1	0	1620		2520	1620	15327	–
acc-2	0	1620		3285	1620	17073	–
acc-3	0	1335		3047	1335	16108	–
acc-4	0	1620		2286	1620	12978	–
acc-5	0	1620		3249	1620	16785	–
acc-6	0	1339		3052	1339	16134	–

Tab. C.2: Dados sobre as instâncias utilizadas por Ghosh (2007).

Instância	Melhor valor	Variáveis binárias	Variáveis contínuas	Linhas	Colunas	Não-zeros
a1c1s1	11505,43	192	3456	3312	3648	10178
a2c1s1	10889,13	192	3456	3312	3648	10178
b1c1s1	24544,25	288	3584	3904	3872	11408
b2c1s1	25740,15	288	3584	3904	3872	11408

Continua na próxima página.

Tab. C.2 – Continuação da página anterior.

Instância	Melhor valor	Variáveis binárias	Variáveis contínuas	Linhas	Colunas	Não-zeros
biella1	3065085	6110	1218	1203	7328	71489
danoint	65,67	56	465	664	521	3232
glass4	1,46E+09	302	20	396	322	1815
markshare1	1	50	12	6	62	312
markshare2	1	60	14	7	74	434
mkc	-563,846	5323	2	3411	5325	17038
net12	214	1603	12512	14021	14115	80384
nsrand-ipx	51200	6620	1	735	6621	223261
rail507	174	63009	10	509	63019	468878
seymour	423	1372	0	4984	1372	33549
sp97ar	6,62E+08	14101	0	1761	14101	290968
sp97ic	4,28E+08	12497	0	1033	12497	316629
sp98ar	5,3E+08	15085	0	1435	15085	426148
sp98ic	4,49E+08	10894	0	825	10894	316317
swath	471,03	6724	81	884	6805	34965
tr12-30	130596	360	720	750	1080	2508
berlin-5-8-0	62	794	289	1532	1083	4507
bg512142	189183,2	240	552	1307	792	3953
blp-ic97	4048,35	9753	92	923	9845	118149
blp-ic98	4494,68	13550	90	717	13640	191947
blp-ar98	6211,45	15806	215	1128	16021	200601
cms750-4	253	7196	4501	16381	11697	44903
dc1c	1843531	8380	1659	1649	10039	121158
dc1l	1813853	35638	1659	1653	37297	448754
dg012142	2706924	640	1440	6310	2080	14795
railway-8-1-0	400	1177	619	2527	1796	7098
trento1	5190144	6415	1272	1265	7687	93571
usabbrv-8-25-70	121	1681	631	3291	2312	9628
aflow40b	1168	1364	1364	1442	2728	6783
dano3mip	688,26	552	13321	3202	13873	79655
ds	413,78	67732	0	656	67732	1024059
fast0507	174	63009	0	507	63009	409349
harp2	-7,4E+07	2993	0	112	2993	2993
liu	1212	1089	67	2178	1156	1156
t1717	216557	73885	0	551	73885	325689

Apêndice D

Resumo dos Testes de Wilcoxon do Capítulo 5

Este apêndice apresenta tabelas contendo algumas informações adicionais relativas aos testes de Wilcoxon discutidos no Capítulo 5. Nessas tabelas, "Variante A" e "Variante B" representam duas versões diferentes da busca tabu paramétrica ou as heurísticas do Cplex. A coluna "Empates" indica em quantas instâncias as duas variantes encontraram o mesmo valor de função objetivo. A coluna referente ao teste de Wilcoxon indica qual versão domina a outra ou se o resultado foi inconclusivo. Por exemplo, na Tabela D.1, o teste de Wilcoxon envolvendo as variantes ACVO e DM revela que a ACVO domina a DM, ocorrem 12 empates, a execução DM consegue resultados melhores em 9 instâncias e a ACVO alcança resultados melhores em 36 instâncias. No título de cada tabela deste apêndice é citada a tabela de referência presente no Capítulo 5.

Tab. D.1: GAP – Cálculo da penalidade inteira e da resistência a metas – Tabela 5.1.

Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
ACVO	36	DM	9	12	Variante A
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
ACVO	39	RB	8	10	Variante A
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
DM	22	RB	22	13	Inconclusivo

Tab. D.2: MKP – Cálculo da penalidade inteira e da resistência a metas – Tabela 5.3.

Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
ACVO	17	DM	11	4	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
ACVO	11	RB	17	4	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
DM	9	RB	19	4	Inconclusivo

Tab. D.3: MDMKP – Cálculo da penalidade inteira e da resistência a metas – Tabela 5.5.

Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
ACVO	15	DM	10	7	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
ACVO	13	RB	12	7	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
DM	12	RB	12	8	Inconclusivo

Tab. D.4: GAP – Resultados de curto prazo – Tabela 5.7.

Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO(1^{st})	4	HCPLEX	50	3	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO(1^{st})	0	BTP-ACVO	42	15	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	17	HCPLEX	36	4	Inconclusivo

Tab. D.5: MKP – Resultados de curto prazo – Tabela 5.9.

Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO(1^{st})	2	HCPLEX	30	0	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO(1^{st})	0	BTP-ACVO	26	6	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	12	HCPLEX	20	0	Variante B

Tab. D.6: MDMKP – Resultados de curto prazo – Tabela 5.11.

Variante A		Variante B		Empates	Teste de Wilcoxon
BTP-ACVO(1^{st})	15	HCPLEX	12	5	Variante A
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO(1^{st})	0	BTP-ACVO	25	7	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	20	HCPLEX	7	5	Variante A

Tab. D.7: GAP – Resultados de extensões do curto prazo – Tabela 5.13.

Variante A		Variante B		Empates	Teste de Wilcoxon
BTP-ACVO	30	BTP-Cortes	16	11	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	34	BTP-IM-B&C	11	12	Variante A
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	26	BTP-RFO	5	26	Variante A
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	22	BTP-ACVO (*Probing*)	26	9	Inconclusivo

Tab. D.8: MKP – Resultados de extensões do curto prazo – Tabela 5.15.

Variante A		Variante B		Empates	Teste de Wilcoxon
BTP-ACVO	21	BTP-Cortes	3	8	Variante A
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	10	BTP-IM-B&C	12	10	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	17	BTP-RFO	7	8	Variante A
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	17	BTP-ACVO (*Probing*)	6	9	Variante A

Tab. D.9: MDMKP – Resultados de extensões do curto prazo – Tabela 5.17.

Variante A		Variante B		Empates	Teste de Wilcoxon
BTP-ACVO	26	BTP-Cortes	1	5	Variante A
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	16	BTP-IM-B&C	10	6	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	20	BTP-RFO	6	6	Variante A
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	17	BTP-ACVO (*Probing*)	8	7	Variante A

Tab. D.10: GAP – Resultados de longo prazo – Tabela 5.19.

Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	10	MF_DIV_INT	18	20	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	6	*RK*_2–CD_MAX_CC	23	28	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	5	*RK*_1–SDCV	28	24	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	8	*RK*_1–SDCV (*Probing*)	24	25	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
MF_DIV_INT	3	*RK*_1–SDCV	21	24	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_2–CD_MAX_CC	2	*RK*_1–SDCV	17	38	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
MF_DIV_INT	12	*RK*_1–SDCV (*Probing*)	28	8	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_2–CD_MAX_CC	15	*RK*_1–SDCV (*Probing*)	34	8	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_1–SDCV	25	*RK*_1–SDCV (*Probing*)	24	8	Variante A

Tab. D.11: MKP – Resultados de longo prazo – Tabela 5.21.

Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	7	MF_DIV	12	13	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	6	*RK*_1–CD_MAX_CC–CI	16	10	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	0	*RK*_1–SDCV	24	8	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	4	*RK*_1–SDCV–RB	23	5	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
MF_DIV	1	*RK*_1–SDCV	24	7	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_1–CD_MAX_CC–CI	0	*RK*_1–SDCV	25	7	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
MF_DIV	5	*RK*_1–SDCV–RB	21	6	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_1–CD_MAX_CC–CI	3	*RK*_1–SDCV–RB	22	7	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_1–SDCV	12	*RK*_1–SDCV–RB	8	12	Inconclusivo

Tab. D.12: MDMKP – Resultados de longo prazo – Tabela 5.23.

Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	13	MF_DIV	12	7	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	7	*RK*_1–CD_MAX_SC	20	5	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	0	*RK*_1–SDCV	29	3	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
BTP-ACVO	1	*RK*_1–SDCV–DM	27	4	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
MF_DIV	0	*RK*_1–SDCV	28	4	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_1–CD_MAX_SC	1	*RK*_1–SDCV	26	5	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
MF_DIV	1	*RK*_1–SDCV–DM	26	5	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_1–CD_MAX_SC	1	*RK*_1–SDCV–DM	25	6	Variante B
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_1–SDCV	12	*RK*_1–SDCV–DM	11	9	Inconclusivo

Tab. D.13: Execuções de MDMKP pelo Cplex e pela BTP – Tabela 5.25.

Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
CPLEX-2h	15	*RK*_1–SDCV	9	8	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
CPLEX-2h	13	*RK*_1–SDCV–2h	11	8	Inconclusivo
Variante A		**Variante B**		**Empates**	**Teste de Wilcoxon**
*RK*_1–SDCV	0	*RK*_1–SDCV–2h	16	16	Variante B

Apêndice E

Artigo Submetido ao Periódico Computers and Operations Research

Artigo submetido em
19/07/2009
manuscrito COR-D-09-00564
Revisado em 15/03/2010
Aceito em 08/07/2010

Digital Object Identifier:
doi:10.1016/j.cor.2010.07.004

A computational study of parametric tabu search for 0–1 mixed integer programs

Luís Henrique Sacchi
Escola de Engenharia de Piracicaba
Av. Monsenhor Martinho Salgot, 560,
CEP 13414-040, Piracicaba, SP, Brazil
sacchi@densis.fee.unicamp.br

Vinícius Amaral Armentano
Faculdade de Engenharia Elétrica e de Computação,
Universidade Estadual de Campinas
Av. Albert Einstein, 400,
CEP 13083-952, Campinas, SP, Brazil
vinicius@densis.fee.unicamp.br

July 21, 2010

Abstract

We present a computational study of parametric tabu search for solving 0–1 mixed integer programming (MIP) problems, a generic heuristic for general MIP problems proposed by Glover [Glover, F., Parametric tabu-search for mixed integer programs, Computers and Operations Research, 33, 2449-2494, 2006]. This approach solves a series of linear programming problems by incorporating branching inequalities as weighted terms in the objective function. New strategies are proposed for uncovering feasible and high-quality solutions and extensive computational tests are performed on instances from the literature.

Keywords: Mixed integer programming, Penalized branching, Tabu search, Heuristics

1 Introduction

The 1990's is the decade where the use of mixed integer programming (MIP) to model combinatorial problems changed and increased dramatically due to the increase of hardware capacity in personal computers and workstations, and the impressive research and implementation dedicated to the design of commercial general-purpose solvers. The new versions of MIP commercial and open source solvers allow the user to have a greater control on parameter settings that yield better performance for solving a specific model. However, in some complex cases, a general-purpose solver may not be an adequate choice, and one tends to develop a heuristic, thus losing the advantage of working in a generic and well-explored framework. Recently, general-purpose local search heuristics for solving MIP problems have been proposed in the literature such that they interact with a MIP-solver in order to quickly find a high quality solution or the first feasible solution. This approach allows a promising integrated way for obtaining multiple high quality solutions in a reasonable time.

Fischetti and Lodi [14] develop a procedure that uses local search in the neighborhood of the best current solution for solving a MIP with 0–1 variables. The neighborhood is obtained by the addition in the MIP of general linear inequalities called local branching cuts. This type of cut allows a soft variable fixing strategy in the sense that a local branching cut divides the space into two disjoint subspaces, such that solutions in one subspace contains binary vectors that differ from the best current solution by at most k components and the solutions in the other subspace differ from such a solution by at least $k+1$ components. The soft variable fixing contrasts with the hard variable fixing (diving), in which some 0–1 variables are fixed and a smaller MIP is solved. Hansen et al. [24] apply variable neighborhood search to local branching by increasing the neighborhood size, i.e., starting from a value $k_{\min}$ and increasing the value of k without limit. This size is reduced to $k_{\min}$ either when the current best solution is updated or no feasible solution is found in a node within a time limit.

Danna et al. [10] propose a local search for a MIP called Relaxation Induced Neighborhood Search (RINS), which at a node of the branch-and-cut tree fixes the variables that have the same values in the best known solution and in the current LP relaxation. An objective cutoff based on the objective value of the best known solution is included in the model and a sub-MIP on the remaining variables is solved by a solver. The rationale for this method is that the best known solution is feasible with respect to the integrality constraints but it may not be optimal. On the other hand, the solution of the LP relaxation at the current node is in general not integral,

but its objective value is always better than that of the best known solution. Ghosh [22] combines the ideas of local branching and RINS by performing a local search in the region defined by the sum of distances between values of the variables of the LP solution at the current node and the values of the variables of the best known solution.

Fischetti et al. [13] develop a heuristic called Feasibility Pump (FP) with the objective of quickly find a first feasible solution for hard MIPs. The process starts with the optimal solution of the LP relaxation and the corresponding rounded solution which is in general infeasible relative to the constraints. Then an LP that minimizes the distance between these two solutions subject to the MIP constraints is solved and its solution is rounded, thus generating a sequence of LPs that in general converges, i.e., the LP solution is integer. Bertacco et al. [9] suggest an improved implementation of FP for MIPs with general integer variables. Achterberg and Berthold [1] propose a variation of the FP for a MIP by minimizing a convex combination of the objective function and the distance between the LP solution and its rounded solution. Fischetti and Lodi [15] suggest a hybrid heuristic for a MIP that combines FP with local branching. In the first phase, the FP is applied for a very short time to obtain an integer solution which is not feasible relative to the MIP constraints. Artificial values are then added in the violated constraints of the MIP model, and in the spirit of phase I of the simplex method, the sum of the artificial variables is minimized subject to the modified MIP constraints. Other approaches to general-purpose heuristic include [5, 6, 7, 11, 12, 18, 19, 20, 25, 26, 29, 30, 34], among others.

In this paper we undertake a computational study of parametric tabu search for solving 0–1 MIP problems, a generic heuristic for general MIP problems proposed by Glover [17] that solves a series of linear programming problems incorporating branching inequalities as weighted terms in the objective function. The approach extends and modifies a parametric branch-and-bound [16], by replacing its tree search memory by the adaptive memory framework of tabu search [21] that provides greater flexibility and facilitates the use of strategies outside the scope of tree search. The remainder of this paper is organized as follows. Closely following Glover [17], sections 2 and 3 describe the components of the core parametric tabu search method and an associated algorithm. Our main contribution lies in section 4, where we propose alternative strategies for finding feasible and high-quality solutions, and in Section 5 which contains computational experiments involving parameters setting, selection of strategies and comparative results with the objective feasibility pump [1] and CPLEX 10's heuristics [27] for instances from the literature. Conclusions and analyses are outlined in Section 6.

2 Principles of parametric tabu search

The 0–1 mixed integer program is represented as

$$\begin{array}{lll} \text{(MIP)} & \text{Minimize} & x_0 = cx + dy \\ & \text{subject to} & (x, y) \in W \\ & & x \in X \end{array}$$

where

$$W = \{(x, y) : Ax + Dy \geq b, e \geq x \geq 0\},$$
$$X = \{e \geq x \geq 0 \text{ and } x \text{ integer}\},$$

A and D are matrices of dimensions $(m \times n)$ and $(m \times p)$, respectively, e denotes a vector with all components equal to 1, x is the vector of binary variables and y represents the vector of continuous variables. Assume that the inequalities $Ax + Dy \geq b$ include an objective function constraint $cx + dy \leq x_0^* - \epsilon$, such that x_0^* is the current best known solution value to (MIP) and ϵ is a small positive number.

The linear programming relaxation of (MIP), which contains only the restriction $(x, y) \in W$, will be denoted by (LP). A vector (x, y) is said to be LP feasible if it is feasible for (LP), and a vector x is integer feasible if the components of x are integers, and hence a (MIP) feasible solution is one that is LP feasible and integer feasible.

Let N^+ and N^- denote selected subsets of $N = \{1, 2, \cdots, n\}$, the index set for x. Consider the set $N' = N^+ \cup N^-$, and let $x' \in X$ be a trial solution such that its components are $x'_j, j \in N'$, and the remaining components $j \in N - N'$ are disregarded.

As in parametric branch-and-bound, the parametric tabu search attempts to impose the following conditions:

$$\begin{array}{llll} x_j \geq x'_j, & j \in N^+ & (\text{if } x'_j = 1) & \text{(UP)}, \\ x_j \leq x'_j, & j \in N^- & (\text{if } x'_j = 0) & \text{(DN)}. \end{array}$$

The conditions (UP) and (DN) represent *goal conditions* and x'_j is called the *goal value*. Such conditions are not enforced directly as in branch-and-bound but rather indirectly by including them in the objective function of the problem (LP). The resulting penalized linear programming, in which M_j denotes a positive parameter is given by

$$\begin{array}{lll} \text{(LP}') & \text{Minimize} & u_0 = cx + dy + \sum\limits_{j \in N^-} M_j x_j + \sum\limits_{j \in N^+} M_j (1 - x_j) \\ & \text{subject to} & (x, y) \in W. \end{array}$$

The problem (LP′) is said to target the conditions (UP) and (DN). A two-phase approach is used to solve (LP′), by first creating a primary objective that disregards the component $cx+dy$. In the optimal solution of this phase, the non-basic variables are fixed at their current assigned binary values. In the second phase, the component $cx + dy$ is minimized over the residual constraints.

A simple approach for the first phase consists of setting $M_j = 1$, for all $j \in N'$. In a more elaborate approach [17], the weight has more emphasis on recent goals and then it decays exponentially according to the number of iterations, as expressed below:

$$M_j = [(1+r)^{NI-(Iter-GI_j)}], \quad \text{if } NI-(Iter-GI_j) \geq 0; \quad 1 \text{ otherwise,} \quad (1)$$

where GI_j denotes the iteration where a goal for the variable x_j is established, $Iter$ the current iteration, NI a parameter that indicates the number of iterations that this memory lasts, r a parameter that defines the magnitude of the weight M_j during NI, and the operator $[\cdot]$ rounds a real number to its nearest integer number.

The parametric tabu search method starts with an instance of (LP′) associated with the original linear programming relaxation, when N' is empty. An optimal solution of an instance of (LP′) is represented by (x'', y'') and as we are interested in the optimal values of the binary vector x'', we refer to x'' as the solution of (LP′), with the understanding that y'' is implicit. The parametric TS method proceeds by using information from the solution to (LP′) and an associated new instance (LP′), which corresponds to the next iteration.

2.1 (LP′) Transitions

The transition from one instance of (LP′) is based on rules that define a new goal value x'_j as one of the values $\lfloor x''_j \rfloor$ and $\lceil x''_j \rceil$. There are three types of transitions (T-UP), (T-DN), and (T-FREE), expressed as follows:

(i) Set $x'_j := \lfloor x''_j \rfloor + 1$ and add j to N^+ (to target) $x_j \geq x'_j$ (T-UP).
(ii) Set $x'_j := \lceil x''_j \rceil - 1$ and add j to N^- (to target) $x_j \leq x'_j$ (T-DN).
(iii) Remove j from N' (to release x_j from (UP) and (DN)) (T-FREE).

The execution of these transitions depends on two types of conditions, called *goal infeasibility* and *integer infeasibility*, described next.

2.2 Goal infeasibility

An optimal solution $x = x''$ to (LP′) is said to be goal infeasible if it violates a current goal condition (UP) or (DN), i.e.

(i) for some $j \in N^+, x''_j < x'_j$ (V-UP).
(ii) for some $j \in N^-, x''_j > x'_j$ (V-DN).

A variable associated with a violation (V-UP) or (V-DN) is called a *goal infeasible variable*, and let $G = \{j \in N' : x_j$ is goal infeasible$\}$. The primary goal response to such an infeasibility consists of defining new goals in the opposite direction for a selected subset $G_p \subseteq G$ such that $\{G_p = j \in N' :$ primary response (R-DN) or (R-UP) is executed$\}$, i.e.,

(i) If $x''_j < x'_j, j \in N^+$, transfer j from N^+ to N^- and set $x'_j = \lceil x''_j \rceil - 1$ (R-DN).
(ii) If $x''_j > x'_j, j \in N^-$, transfer j from N^- to N^+ and set $x'_j = \lfloor x''_j \rfloor + 1$ (R-UP).

In addition to the primary responses, we define a secondary goal response that consists of freeing a goal infeasible variable that belongs to set G. Correspondingly, a selected subset $G_s \subseteq G$ is such that $G_s = \{j \in G :$ response (R-FREE) is executed, i.e., remove j from $N'\}$. A measure called *goal resistance* GR_j(UP) or GR_j(DN) of each variable $x_j, j \in G$ represents the amount of violation (V-UP) or (V-DN) resists the imposition of the associated goal condition (UP) or (DN). Since a higher resistance value causes a higher impact value on the objective function of (LP′), we then choose the g_p variables in G_p with largest goal resistance and the g_s variables in G_s are those with largest goal resistance over $G - G_p$. In the presence of goal infeasibility we always choose $g_p \geq 1$, but we may have $g_s = 0$. Goal resistance measures are discussed in section 4.

2.3 Potential goal infeasibility

A variable x_j is called *potentially goal infeasible* if it is goal feasible, but a limited change in M_j causes x_j to become goal infeasible. In order to deal with such variables, a measure of goal resistance related to the decrease in M_j is given by $\mathrm{GR}^0_j = -\mathrm{RC}_j$, where RC_j is the LP reduced cost for the variable x_j associated with an optimal solution (LP′). At LP optimality, $\mathrm{RC}_j \geq 0$ for all $j \in N'$ and $\mathrm{RC}_j = 0$ for all x_j with overt goal resistance.[1] Goal infeasible variables are considered more important than potentially goal infeasible variables, and the goal resistance values GR_j of the first variables are larger than the goal resistance GR^0_j of the latter. Potentially goal infeasible variables are identified by sorting all goal feasible variables in decreasing value of GR^0_j and then selecting the first T^0 variables to be admitted as potentially goal infeasible. The same primary and secondary responses for goal infeasible variables apply to variables that are potentially goal infeasible.

[1] A variable x_j that is non-basic at its upper bound value 1 has a non-positive reduced cost, but by convention we may refer to the non-negative reduced cost of the non-basic slack variable $s_j = 1 - x_j$.

2.4 Integer infeasibility

Let $F = \{j \in N : x_j = x''_j \text{ is fractional}\}$, the set of variables that are *integer infeasible*. A variable x_j is called unrestricted fractional variable if is fractional but does not have a goal. Such variables are defined by the set $D = F - G$. There is no specific primary transition for the case of integer infeasibility, and we may select a preferred transition (T-UP) or (T-DN). The relative preference for an unrestricted variable x_j is based on a choice preference measure CP_j that depends on the up penalty $\text{IP}_j(\text{UP})$ and the down penalty $\text{IP}_j(\text{DN})$, as described in section 4. For $D_0 \subseteq D$, the $d_0 = |D_0|$ variables with largest values of CP_j are selected to have goals associated with the following responses: if $\text{IP}_j(\text{DN}) \leq \text{IP}_j(\text{UP})$, $j \in D_0$, execute (T-DN), otherwise execute (T-UP).

3 Tabu search

Next we describe the basic elements of tabu search that are used in our procedure to guide the processes described above.

3.1 Tabu conditions and tabu tenure

In the following we allow (R-DN) and (R-UP) to refer also to the responses (R-DN^0) and (R-UP^0). A tabu restriction is associated to an (R-DN) or (R-UP) response for a given variable, by forbidding the response from being executed, if the opposing response was executed within the most recent tabu tenure iterations. Let $\text{TabuTenure}_j(\text{UP})$ and $\text{TabuTenure}_j(\text{DN})$ denote the tabu tenure of a variable x_j if the restriction was activated by an (R-DN) or (R-UP), respectively.

Let α denote the condition UP or DN and β denote the opposite condition. When the reaction (R-α) is triggered, the reaction (R-β) becomes tabu for a number of iterations that is selected from the interval [$\text{TabuTenure}_j\beta$, MaxTenure] with uniform distribution.

3.2 Aspiration by resistance

The aspiration criterion in tabu search allows a tabu response to be released from a tabu restriction if the response has special merit. In the present context, we consider the aspiration by resistance, based on the greatest resistance a specific response has generated in the past. Let $\text{Aspire}_j(\text{DN})$ and $\text{Aspire}_j(\text{UP})$ denote the largest goal resistance $\text{GR}_j(\text{DN})$ and $\text{GR}_j(\text{UP})$ that occurred for x_j on any iteration in which this variable was chosen to execute

an (R-UP) or (R-DN). The tabu restriction for an (R-UP) response is disregarded if GR_j (DN) $>$ Aspire_j(UP). Analogously, the tabu restriction for an (R-DN) response is disregarded if GR_j (UP) $>$ Aspire_j(DN).

A response is called *admissible* if it is either not tabu or else satisfies the aspiration, and is called inadmissible, otherwise. If the response for a goal infeasible variable is inadmissible, then the variable is not permitted to enter the sets G_p and G_s. The only exception to this rule is when G_p is empty. In this case the aspiration by default that allows G_p to contain a variable with a smallest remaining tabu tenure is used.

The overt goal infeasible variables associated to Aspire_j also dominate the potentially goal infeasible variables associated to Aspire_j^0, defined by GR_j^0.

3.3 Determining the values of $\mathrm{TabuTenure}_j$

The tabu tenure of a variable is determined from the values of the minimum, maximum and incremental tabu tenures, which are proposed in this work. The $\mathrm{TabuTenure}_j\alpha$ is then updated according to the following scheme based on Glover [17]:

1. $\mathrm{TabuTenure}_j\alpha$ starts at the minimum value MinTenure $= k_1 \times m$, where m is the number of constraints of (LP′), k_1 is a parameter such that $0 < k_1 \leq 1$, and $\mathrm{Aspire}_j := -\infty, j \in N$.

2. Whenever x_j is subjected to an (R-α) response, because of a (V-β) violation, increase $\mathrm{TabuTenure}_j\beta$ by the increment Δ Tenure $= \max(1, k_2 \times n)$, where n is the number of binary variables and k_2 is a parameter such that $0 < k_2 \leq 1$, and setting $\mathrm{TabuTenure}_j\beta = \mathrm{TabuTenure}_j\beta + \Delta$ Tenure. The value of $\mathrm{TabuTenure}_j\beta$ is not altered if the (R-α) response is tabu, but is executed because of satisfying the criterion of aspiration by resistance. The value of $\mathrm{TabuTenure}_j\beta$ cannot exceed MaxTenure $= k_3 \times m$, where k_3 is a parameter such that $0 < k_1 \leq k_3 \leq 1$.

3. After ClearMaxIter iterations and each time a new best solution is obtained, re-start the process by setting all tabu tenures to MinTenure and all Aspire_j to $-\infty$.

3.4 Core parametric tabu search

The core method starts by specifying (LP′) to be the original relaxation (LP) of (MIP), where no goal conditions exist and N' is empty. It then proceeds as indicated by the flowchart in Figure 1. At the beginning of each iteration (LP′) is solved to obtain an optimal solution (x'', y''). If this solution is (MIP) feasible, update x_0^* and re-optimize (LP′). If (LP′) has no feasible solution, the method stops and the best solution is optimal. If the

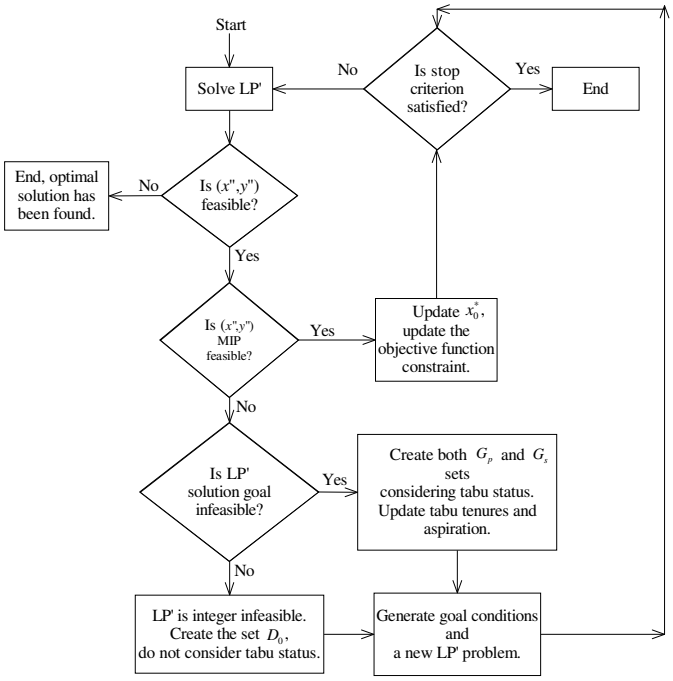

Figure 1: Flowchart for the core parametric tabu search

current solution (x'', y'') is not (MIP) feasible, then it is either goal infeasible or integer infeasible. If it is goal infeasible, create the sets G_p and G_s to consist of the g_p and g_s highest ranking goal infeasible (and potentially goal infeasible) admissible variables from G, i.e., those variables that are not not tabu or else satisfy the aspiration criterion. The associated tabu tenures and aspiration values are updated and the goals of the variables from G_p are set in the opposite direction and the variables from G_s are released from their goals. If the current solution is integer infeasible, create the set D_0 which contains the highest ranking unrestricted fractional variables from D that receive goals. In any case, new goals are established for the construction of a new (LP$'$). If the stopping criterion is reached, the method stops, otherwise, a new iteration is started.

4 Measures and cardinality

In this section we present three measures for goal resistance GR_j and choice preference CP_j associated with a variable x_j, as well as procedures to define the cardinality of sets G_p, G_s and D_0.

4.1 Measures for goal resistance and choice preference

At the simplest version [17], $\mathrm{GR}_j = |x''_j - x'_j|$ that identifies how distant is the (LP′) value x''_j from its current goal value x'_j. For the integer penalty, we first compute the fractions $f_j^+ = \lceil x''_j \rceil - x''_j$ and $f_j^- = x''_j - \lfloor x''_j \rfloor$, and then we set $\mathrm{IP}_j(UP) = f_j^+$ and $\mathrm{IP}_j(DN) = f_j^-$. The choice preference is expressed as

$$\mathrm{CP}_j^1 = (\mathrm{IP}_j(\mathrm{UP}) + \mathrm{IP}_j(\mathrm{DN}))(|\mathrm{IP}_j(\mathrm{UP}) - \mathrm{IP}_j(\mathrm{DN})| + w)$$

where w is a small positive weight to give additional influence to the sum of the penalties, as in the case when their absolute difference is zero.

When a new (LP′) is defined, every new added goal that is satisfied causes an increase in the objective function of (LP′). Thus, at a more elaborate level, the goal resistance GR_j can be defined as an estimate of the increase in the objective of (LP′). We suggest the use of the reliability branching estimate proposed by Achterberg et al. [3] that combines the mechanisms of pseudocost branching [8] and strong branching [28, 4]. At the beginning of the search the upward ψ_j^+ and downward ψ_j^- pseudocosts of a variable are not reliable due to the lack of information, and in this case strong branching is applied to every variable that takes on a fractional value until its pseudocost becomes reliable. At this point, the pseudocost for each variable is fixed, thus avoiding the use of post-optimization dual simplex in (LP′). Reliability branching was applied with success to select branching variables in the cut-and-branch method for solving instances from the literature MIPLIB 2003 [2] and instances used by Mittelmann [31] and for this reason this technique is used to compute measures of goal resistance and integer penalty. Recall that a two-phase approach is used to solve (LP′), and at the end of the first phase the non-basic variables with satisfied established goals are fixed to their binary values. Let (LP'_R) denote the residual problem over the basic variables and non-basic variables without goals, and let Q_j^- and Q_j^+ represent the subproblems derived from (LP'_R) when the constraints $x_j \leq \lfloor x''_j \rfloor$ and $x_j \geq \lceil x''_j \rceil$ are imposed, respectively, and consider the fractions f_j^+ and f_j^-. We then apply reliability branching to those basic variables that do not have reliable pseudocosts as described in the following.

If a limit of γ iterations of the dual simplex method is used as a post-optimization to obtain the sub-optimal objective function values $z_{Q_j^-}$ and

$z_{Q_j^+}$ for the subproblems Q_j^- and Q_j^+, the selection of branching variables is called strong branching, and if the subproblems Q_j^- and Q_j^+ are solved to optimality, the selection is called full strong branching. Let $\Delta_j^- = z_{Q_j^-} - z_R$ and $\Delta_j^+ = z_{Q_j^+} - z_R$ represent the difference between such values and the optimal objective function of (LP$'_R$). Let η_j^- and η_j^+ be the number of times that a variable x_j was selected in previous iterations as a branching variable and that the resulting subproblems $Q_{j,k}^-$ and $Q_{j,k}^+$ are feasible, respectively. The pseudocosts of a variable x_j are defined as $\psi_j^+ = \{\sum_{k=1}^{\eta_j^+}(\Delta_{j,k}^+/f_{j,k}^+)\}/\eta_j^+$ and $\psi_j^- = \{\sum_{k=1}^{\eta_j^-}(\Delta_{j,k}^-/f_{j,k}^-)\}/\eta_j^-$. Correspondingly, the associated up and down penalties are given by $\mathrm{IP}_j(\mathrm{UP}) = f_j^+\psi_j^+$ and $\mathrm{IP}_j(\mathrm{DN}) = f_j^-\psi_j^-$.

The integer penalties are given by $\mathrm{IP}_j(\mathrm{UP}) = f_j^+\psi_j^+$ and $\mathrm{IP}_j(\mathrm{DN}) = f_j^-\psi_j^-$, and the choice preference is given by the expression [11]

$$\mathrm{CP}_j^2 = (1-\mu)\min\{\mathrm{IP}_j(\mathrm{UP}), \mathrm{IP}_j(\mathrm{DN})\} + \mu\max\{\mathrm{IP}_j(\mathrm{UP}), \mathrm{IP}_j(\mathrm{DN})\}$$

where μ is a parameter such that $0 \le \mu \le 1$.

The pseudocosts of a variable x_j are called unreliable if $\min\{\eta_j^-, \eta_j^+\} < \eta_{rel}$, where η_{rel} is the reliability parameter. An outline of the reliability branching for selecting a branching variable when the search is under integer infeasibility is shown in the following algorithm.

Step 0. Consider the set D of integer infeasible variables.

Step 1. Sort the elements $j \in D$ in non-increasing order of the choice preference CP_j^2. For each $j \in D$ with $\min\{\eta_j^-, \eta_j^+\} < \eta_{rel}$, do

a) Execute at most γ iterations of the dual simplex method on each subproblem Q_j^- and Q_j^+, and calculate Δ_j^- and Δ_j^+.

b) Update the pseudocosts ψ_j^+ and ψ_j^- with Δ_j^+ and Δ_j^-.

c) Update the choice preference values CP_j^2.

d) If $\max_{j\in D}\{\mathrm{CP}_j^2\}$ has not changed for λ consecutive updates of CP_j^2, go to step 2.

Step 2. Return the d_0 indices of the set D with the largest values of CP_j.

The reliability branching algorithm for the set G of goal infeasible variables is similar to the above, and we include it for the sake of clarity.

Step 0. Consider the set G of goal infeasible variables.

Step 1. Sort the elements $j \in G$ in non-increasing order of $\mathrm{GR}_j(\mathrm{UP})=f_j^+\psi_j^+$ or $\mathrm{GR}_j(\mathrm{DN})= f_j^-\psi_j^-$. $\mathrm{GR}_j(\mathrm{UP})$ is used if x_j violates a goal conditon UP, (V-UP), and $\mathrm{GR}_j(\mathrm{UP})$ is used if x_j violates a goal conditon DN, (V-DN). For each $j \in G$ with $\min\{\eta_j^-, \eta_j^+\} < \eta_{rel}$, do

a) If x_j is associated to V-UP, execute γ iterations of the dual simplex method on the subproblem Q_j^+ and compute Δ_j^+. Otherwise, perform γ iterations of the dual simplex method on the subproblem Q_j^- and compute Δ_j^-.

b) Update the pseudocost ψ_j^+ with Δ_j^+ or the pseudocost ψ_j^- with Δ_j^-, according to the action taken in Step 1a.

c) Update the goal resistance value $\mathrm{GR}_j(\mathrm{UP})= \Delta_j^+$ or $\mathrm{GR}_j(\mathrm{DN})= \Delta_j^-$ according to Step 1a.

d) If $\max\limits_{j\in G}\{\mathrm{GR}_j(\alpha)\}$, for $\alpha =$UP or DN according to the action taken in Step 1a, has not changed for λ consecutive updates, go to step 2.

Step 2. Return the g_p indices of the set G with the largest values of $\max\limits_{j\in G}\{\mathrm{GR}_j(\alpha)\}$ to form the set G_p and the g_s indices of the set $G-G_p$ with the largest values of $\max\limits_{j\in G-G_p}\{\mathrm{GR}_j(\alpha)\}$ to compose the set G_s.

The third measure for computing the goal resistance GR_j and the integer penalty IP_j is based on the scheme P[2] of the active-constraint variable ordering strategy (ACVO) proposed by Patel and Chinnneck [33] for selecting a branching variable in the branch-and-bound context. This mechanism estimates the impact of the candidate variables on the active constraints in the current LP relaxation, and it aims to quickly find the first feasible solutions to a MIP.

A weight w_{ij} is associated with every binary variable that assumes a fractional value in the active constraint i, and is defined as $w_{ij} = |a_{ij}|/N_i^F$, where a_{ij} is the coeficient of the matrix A and N_i^F is the number of binary variables that takes on fractional values in the active constraint i. Thus, larger values of a_{ij} or smaller values of N_i^F cause a larger impact. If x_j is not present in constraint i or constraint i is not active, w_{ij} is set to zero. In the ACVO strategy the variables with larger sum of weights w_{ij} are chosen. The number of these variables is the sum of g_s and g_p if (LP′) is goal infeasible or d_0 if (LP′) is integer feasible. The application of this approach in the context

[2]The schemes from A to O proposed by proposed by Patel and Chinnneck [33] did not perform well in the context of parametric TS

of parametric TS showed to be more effective when the sum of weights is multiplied by the factor $1/f_j^+$ or $1/f_j^-$, which favors the selection of binary variables that assume values closer to 0 or 1. The integer penalties are thus defined as

$$\mathrm{IP}_j(\mathrm{DN}) = \infty, \quad \mathrm{IP}_j(\mathrm{UP}) = \frac{1}{f_j^+}\sum_{i=1}^{n} w_{ij} \qquad \text{if} \quad f_j^+ \leq f_j^-$$

$$\mathrm{IP}_j(\mathrm{UP}) = \infty, \quad \mathrm{IP}_j(\mathrm{UP}) = \frac{1}{f_j^-}\sum_{i=1}^{n} w_{ij} \qquad \text{if} \quad f_j^+ > f_j^-$$

Similarly, the goal resistance can be expressed as

$$\mathrm{GR}_j(\mathrm{UP}) = \frac{1}{f_j^+}\sum_{i=1}^{n} w_{ij} \quad \text{and} \quad \mathrm{GR}_j(\mathrm{DN}) = \frac{1}{f_j^-}\sum_{i=1}^{n} w_{ij}$$

and the choice preference measure is computed as

$$\mathrm{CP}_j^3 = \mathrm{IP}_j(\mathrm{UP}) \qquad \text{if} \quad f_j^+ \leq f_j^-$$

$$\mathrm{CP}_j^3 = \mathrm{IP}_j(\mathrm{DN}) \qquad \text{if} \quad f_j^+ > f_j^-$$

4.2 Cardinality of sets

A simple scheme for determining the cardinality of the sets G_p, G_s and D_0 is to define suitable positive fractional parameters f_p, f_s, f_d, and then set $d_0 = |D_0| = f_d \times |D|, g_p = |G_p| = f_p \times |G|, g_s = |G_s| = f_s \times |G|$ throughout the iterations. This is a static scheme that does not take into account the search evolution. The following adaptive strategy here proposed is more elaborate and defines the cardinality of the sets according to the condition of the region that is being visited. The idea of the algorithm shown below is to seek to solve the goal infeasibility condition by inverting a small number of goals and if this infeasibility persists the emphasis is shifted to free a large number of variables with unsatisfied goals. Consider the parameters $\delta_2 > 1,\ 0 < \delta_1 < 1,\ 0 < k \leq 1,\ 0 < f_s < 1$, the maximum value $0 < f_{d\max} \leq 1$ for the fractional parameter f_d, and the extreme set cardinality values $g_{\max}, g_{p\min}, g_{s\min}$. Let $Iter$ denote the current iteration and $IterTrans$ the iteration where the search transits from the integer infeasibility condition to the goal infeasibility condition or when the search remains for $k \times m$ consecutive iterations in one of these conditions. The dynamic cardinality of sets G_p, G_s and D_0 is defined by the following algorithm.

Step 0. *Initialization.* $g_p \leftarrow g_{p\min}, g_s \leftarrow g_{s\min}, IterTrans \leftarrow 0, Iter \leftarrow 0$, initial condition: integer infeasibility

Step 1. *Search is in the integer infeasibility condition.* If $(Iter - IterTrans) > k \times m$ then $f_d \leftarrow \min(f_{d\max}, f_d \times \delta_2)$, and $IterTrans \leftarrow Iter$. Go to step 3.

Step 2. *Search is in the goal infeasibility condition.* If $(Iter - IterTrans) > k \times m$, and if $(g_p > g_{\max})$ or $(g_s > g_{\max})$ then $g_p \leftarrow 1$ and $g_s \leftarrow \max(g_{\max}, f_s \times |G|)$. Else, increase alternatively the parameters g_p and g_s by one unit.

Step 3. *Construction of a new* (LP′). $Iter \leftarrow Iter + 1$. Create new goals, update the tabu list if the current (LP′) is goal infeasible, and solve the new (LP′). If the current (LP′) is integer infeasible and the new (LP′) is goal infeasible, set $f_d \leftarrow f_d \times \delta_1, g_p \leftarrow g_{p\min}$ and $g_s \leftarrow g_{s\min}$ and go to step 2, else go to step 1.

In step 1, the fraction of unrestricted fractional variables is defined as $f_d \leftarrow \min(f_{d\max}, f_d \times \delta_2)$ and as long as $f_d \leq f_{d\max}$, this parameter is slightly increased, for example, $\delta_2 = 1.05$ at every $k \times m$ consecutive iterations in the integer infeasibility condition. Since $0 < \delta_1 < 1$, the fraction f_d is decreased in step 3 according to $f_d \leftarrow (f_d \times \delta_1)$, and this reduction always takes place when the search changes from the integer infeasibility condition to the goal infeasibility condition. The reasoning of this policy is to establish several goals at each iteration, while the search remains in the integer infeasibility condition. When the search transits from the integer infeasibility condition to the goal infeasibility condition, the fraction f_d is substantially reduced by setting, for example, $\delta_1 = 0.5$.

Step 2 deals with the goal infeasible condition and the search tries to resolve conflicts, by altering the minimum number of established goals. Tipically, $g_{p\min} = 1$ and $g_{s\min} = 0$, which implies that one goal is changed and no goals are removed. At every $k \times m$ consecutive iterations g_p and g_s, in that order, are alternately increased by one unit. If g_p or g_s reaches the $g_{\max}$ and the goal infeasibility condition still persists after the following $k \times m$ iterations, the objective is to move a large number of variables from their goal condition to the free condition and hence $g_s \leftarrow \max(g_{max}, f_s \times |G|)$ and $g_p \leftarrow 1$. Every time the search leaves the integer infeasilibity condition and enters the goal infeasibility condition, g_p and g_s are reset, respectively, to the initial values $g_{p\min}$ and $g_{s\min}$ in step 3, where a new (LP′) is constructed.

5 Computational experiments

This section describes the setting of parameters, the selection of strategies for the parametric TS and comparative results with the objective feasibility pump [1] and heuristics from CPLEX 10. The algorithms of the core parametric TS were coded in C++ by using the version 4.0.2 of the GCC compiler and computational tests were carried out on a PC Intel Pentium IV 3.2 GHz, 3Gbyte RAM with the operating system Linux Fedora 4. We tested our implementation on 78 instances from the MIPLIB [2] and the Mittelmann test set [31], and results are compared with those from the objective feasibility pump [1] and the best results, within a time limit of one hour for each instance, of CPLEX 10's heuristics that are applied after solving the root node. As in [1] we applied the MIP preprocessing of CPLEX 10 prior to running parametric tabu search and we also set a time limit of one hour in all runs of each instance. The instances of (LP′) were solved by CPLEX 10.

5.1 Parameters setting

The weight M_j defined in (1) depends on two parameters. The parameter NI that expresses the number of iterations that the memory of an UP or DN condition lasts for a variable x_j was set to 64. The positive parameter r that establishes the influence of the weight M_j yields good results in the range $[0.1, 0.2]$, with a slightly advantage for the value 0.2, which was adopted. For $r < 0.1$ the weight M_j assumes very close values during $NIter$, whereas $r > 0.2$ results in very large values for M_j in the first iterations after the goal for the variable x_j was established. Figure 2 illustrates the variation of M_j with the number of iterations for three values of r.

For almost all tested instances, the parameter ϵ in the objective function constraint $cx+dy \leq x_0^* - \epsilon$, where x_0^* is the current best known solution value to (MIP), was set to the value of 1. The exceptions are the instances *modglob* and *glass*4, which have very large coefficients in their objective functions. As a result, best solutions have very large objective function values and numerical problems prevent that the value $\epsilon = 1$ influences the current (LP′) solution, in the sense that x_0^* does not become infeasible for (LP′). For these instances the parameter ϵ takes on the values 1000 and 450,000, respectively.

For the determination of the parameters related to the tabu tenure, the sets of values $\{0.05, 0.10,\ 0.15, 0.20\}$ and $\{0, 10, 0.15, 0.20, 0.25, 0.30\}$ were tested for the parameters k_1 and k_3, respectively. Values of $k_1 < 0.05$ cause cycling in the tabu search and for values of $k_3 > 0.10$ it is common that all goal infeasible variables become tabu, which is not desirable in the search.

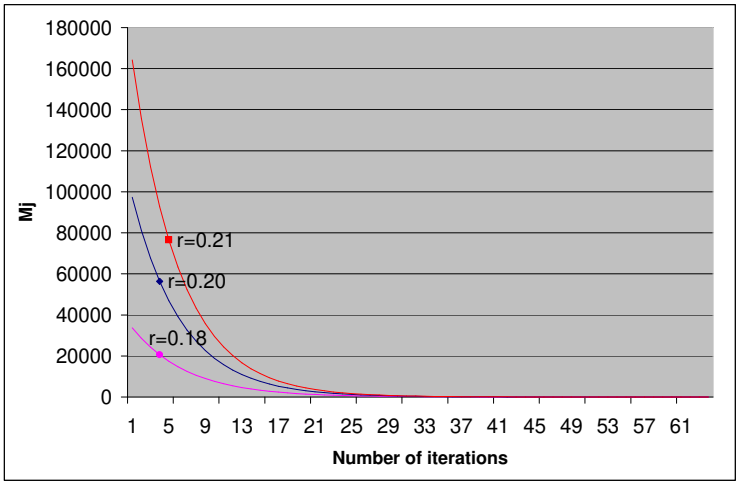

Figure 2: Influence of the parameter r on the weight M_j with the number of iterations

The best values found were $k_1 = 0.05$ and $k_3 = 0.10$. The parameter k_2 should be small in order to avoid that MinTenure reaches MaxTenure prematurely, and the value adopted was 0.01. The parameter ClearMaxIter was set to 2000.

The parameters f_p, f_s and f_d associated with the simple scheme for determining the cardinality of the sets G_p, G_s and D_0 were determined in the following way. The parameter f_d was tested for the values $\{0.05, 0.10, 0.15, 0.20, 0.25, 0.30\}$ and the selected value was 0.05. A value larger than 0.05 is, in general, not appropriate in the case that (LP′) has a large number of goals and enters in the condition of goal infeasibility as a result of new goals generated from an integer infeasibility condition. In some cases, the primary response that changes goals to the opposite direction may solve (LP′). However, in many cases, it is not possible to satisfy the goal conditions for (LP′) and we have to resort to the secondary goal response, which consists of freeing goal infeasible variables. As a result we may end up with a poor (LP′), which is goal feasible but has much less goals than it had before entering the goal infeasibility condition.

The parameter f_p was tested for the same set of values and best results were obtained for the value 0.05. For larger values than this one the inversion of many goals makes it difficult to find integer solutions. In general, the best

option is to invert one or two goals at each iteration. A similar behavior was observed for the parameter f_s, i.e., the release of many variables at any iteration from their goals results in a poor (LP′).

The parameter T^0 of potentially goal infeasible variables was kept at the value of 1 to circumvent the situation in which all overtly goal infeasible variables are in the tabu condition. The empirical analysis of computational tests led to the following parameter values for the adaptive strategy that defines the cardinality of the sets G_p, G_p and $D_0 : g_{p\min} = 1, g_{s\min} = 0, \delta_2 = 1.05, \delta_1 = 0.5, f_{d\max} = 0.3, k = 0.25, g_{\max} = 3$ and $f_s = 0.05$.

The values of the parameters γ, η_{rel} and λ associated with the reliability branching algorithm were set to the values 10, 8 and 8, respectively. The parameter λ restricts the number of iterations of the dual simplex reoptimization, the parameter η_{rel} defines the reliability of the pseudocosts and the parameter λ is the stopping criterion for the algorithm. Large values of η_{rel} and λ lead to a high time consuming algorithm and hence less iterations for the parametric TS are allowed, given that the stopping criterion is 3600 seconds. The parameters w and μ for the choice preference measures CP_j^1 and CP_j^2 were set to the values 0.01 and 5/6, respectively.

5.2 Strategies selection

The non-parametric Wilcoxon signed rank test [32] was applied to compare the expected values $E_X[\mathrm{OV}]$ and $E_Y[\mathrm{OV}]$, where X and Y denote two strategies and OV is a random variable that represents the objective value of a set of instances of a problem. This test is very useful in comparing the performance of two heuristics [23] and consists of computing the difference of the random objective values of OV for each instance. Ranks are assigned to each difference and we test the null hypothesis, $E_X[\mathrm{OV}] = E_Y[\mathrm{OV}]$ with the alternative hypotheses $E_X[\mathrm{OV}] > E_Y[\mathrm{OV}]$ or $E_X[\mathrm{OV}] < E_Y[\mathrm{OV}]$ at the significance level 0.05. If $E_X[\mathrm{OV}] = E_Y[\mathrm{OV}]$, the best strategy is the one that presents a smaller number of instances for which no integer feasible solution was found during the stopping criterion of 3600 seconds. If the tie persists, the best strategy is the one with the least average gap relative to the best known solution value in the literature over all instances, except for the instances *acc*-0 up to *acc*-6 whose optimal solution value is zero.

The first strategic decision is related to the choice of one of the three strategies of measuring goal infeasibility, namely, goal distance (GD), reliability branching (RB) and active-constraint variable ordering (ACVO). For each strategy, Table 1 shows the number of instances and its percentage relative to the total of 78 instances in terms of the gap range such that gap $= 100 \times \frac{x_0^* - x_{best}}{x_{best}}$, where x_{best} is the best known solution value of an instance in

the literature and x_0^* is the best solution value found by each parametric TS version. The table also shows for each strategy the number of failures, which is the number of instances for which the parametric TS was not able to find a feasible solution. For example, for 47 (59.2%) instances the parametric TS with the ACVO strategy yields a gap in the interval from 0% to 5%, and presents 5 failures. The non-classified instance in this table refers to instance acc-5, where the parametric TS obtained a solution with value one and the optimal solution value is zero (see Table 2), which prevents the calculation of the gap. We selected the ACVO strategy, because the Wilcoxon test showed that it is superior to GD and RB strategies. The remaining decisions for the parametric TS were made with the use of the ACVO strategy.

The second decision is related to the effect of the weights M_j in the objective function of (LP′). In one variant M_j is set to the fixed value of 1 for all $j \in N'$ and in another variant M_j is dynamic according to expression (1). The Wilcoxon test showed that the use of the dynamic version in the parametric TS is superior, and for this reason it was selected. In addition, the search with the dynamic variant failed to find integer solutions in 5 instances and has a mean gap of 93% over the best known solutions in the literature, while the fixed version failed in 12 instances and has a corresponding mean gap of 401%. This result shows that the use of the variable weight with exponential decay along the iterations is an important type of memory, because the most recent goals are more likely to be satisfied in earlier iterations than in later iterations.

The third decision is concerned with selecting the fixed or the adaptive approach to define the cardinality of the sets G_p, G_s and D_0. The Wilcoxon was not conclusive to determine the best strategy, and we chose the adaptive approach because it fails in 5 instances, while the static approach fails in 8 instances.

5.3 Comparative results

The methods parametric TS, objective feasibility pump, denoted OFP [1], and CPLEX 10' root node heuristics, represented by HCPLEX were applied to 78 instances with names in the first column of Table 2. Results for parametric TS-first, denoted PTS-first, and objective feasibility pump refer to the first feasible solution found, the time it was obtained and the gap relative to the best known solution values in the literature. Results for parametric TS, denoted PTS-3600, and HCPLEX are obtained by running them for 3600 seconds and then reporting the best solution. Objective values in bold indicate optimal solutions values. A bar '−' means that no solution was found within the time limit of one hour, and a gap equals zero means that the best

known solution has been reached. The symbol '0^+' is used for solution values with gap of less than 0.5%. The bottom of this table displays the mean deviation (Mean) over all instances, the mean deviation over those instances for which all methods found a solution (Mean(57)), and the number of failures (Failures) which is the number of instances for which a method could not find a feasible solution. Therefore, Mean(57) indicates that all methods found a solution for 57 instances. Achterberger and Berthold [1] tested a different version of the instance neos[3] that is now available.

PTS-first obtains better solutions in 32 instances, while OFP yields better results for 38 instances, with 7 ties in instances which both methods could not find a feasible solution. The Wilcoxon test applied to PTS-first and OFP did not establish any dominance, but the mean gap of OFP is 922% is better than that of PTS-first which is 1248%. OFP fails in 4 instances and is successful in the instances cap6000, momentum 1 and rd-rplusc-21 where PTS-first fails, while PTS-first fails in 5 instances and is successful in the instances ds and acc-6 for which OFP and HCPLEX fail. The instance stp3d could not be solved for any method. Regarding computational times, PTS-first is faster in 20 instances, OFP is faster in 28 instances and ties occur for 29 instances.

The Wilcoxon test was also inconclusive when comparing PTS-first and HCPLEX, and the latter presents an average gap of 1909%. PTS-first yields better solutions for 24 instances and HCPLEX obtains better solutions in 46 instances. PTS-first presents a lower Mean(57) compared to HCPLEX, which fails in 16 instances. On the other hand, HCPLEX is faster in 47 instances, slower in 7 instances, with 30 ties.

The Wilcoxon test was also inconclusive when comparing OFP and HCPLEX. OFP obtains better solutions for 30 instances, worse solutions in 40 instances, with 7 ties. HCPLEX is faster in 34 instances, slower in 13 instances, with ties in 30 instances.

The performance of PTS-3600 is superior to the other methods with respect to the Wilcoxon test, which shows its capability of obtaining much better solutions when additional computational time is allowed. For 60 instances it was able to improve the solution obtained by PTS-first, reducing the average gap of 1248% to 93.21%. PTS-3600 produced better solutions for 56 and 55 instances, worse solutions in 12 and 14 instances, equal solution values in 10 and 8 instances, when compared to HCPLEX and OFP. It also finds 21 optimal solutions, whereas PTS-first, OFP and HCPLEX obtain, 8, 10 and 6 optimal solutions, respectively.

Table 3 shows that PTS-3600 find solutions for 47 instances with gap less

[3]private conversation

than 5%, whereas PTS-first, OFP and HCPLEX find 23, 23 and 28 solutions within the same gap interval. Moreover, PTS-3600 obtains fewer number of solutions with gap greater than 50%.

Table 1: Comparison of integer penalty and goal resistance approaches

	ACVO		GD		RB	
Gap range	Number	%	Number	%	Number	%
Less than 1%	32	41.0	30	38.5	28	32.1
1% to 5%	15	19.2	13	16.7	16	14.1
5% to 10%	6	7.7	10	12.8	8	14.1
10% to 25%	5	6.4	4	5.1	6	12.8
25% to 50%	5	6.4	4	5.1	4	1.3
50% to 100%	3	3.8	3	3.8	1	9.0
Greater than 100%	6	7.7	5	6.4	3	5.1
Non-classified	1	1.3	0	0.0	0	0.0
Failures	5	6.4	9	11.5	12	11.5

Table 2: Parametric tabu search compared with objective feasibility pump and CPLEX heuristics

	PTS-first			PTS-3600			OFP			HCPLEX		
Name	Sol. value	Gap %	Time	Sol. value	Gap %	Time	Sol. value	Gap %	Time	Sol. value	Gap %	Time
10teams	1054	14	28	**924**	**0**	**346**	952	3	5	-	-	1
a1c1s1	19328,2	68	17	16757,2	46	20	16076,6	40	1	21029,4	83	1
aflow30a	3100	168	1	1227	6	2031	4105	254	0	1239	7	0
aflow40b	6125	424	17	2358	102	1351	2049	75	0	1439	23	1
air04	56496	1	14	56138	0^+	733	57298	2	164	-	-	6
air05	27485	4	8	26444	0^+	1805	26942	2	8	27291	3	3
cap6000	-2,45E+06	0^+	37	-2,45E+06	0^+	37	-2,43E+06	1	0	-2,45E+06	0^+	0
dano3mip	856,5	20	117	743,9	4	315	769,3	8	383	714,125	0	140
danoint	82	25	0	67	2	2209	87	32	3	-	-	1
disctom	-	-	3600	-	-	3600	**-5000**	**0**	**11**	-	-	6
ds	701,055	70	3313	692,795	68	3315	-	-	3600	412,5025	0	88
fast0507	183	5	36	175	1	981	179	3	21	177	2	32
fiber	567919	40	0	423613	4	1454	1,21E+06	197	0	468924	16	0
fixnet6	5376	35	0	3985	0^+	2594	4807	21	0	4435	11	0
glass4	3,90E+09	225	0	2,70E+09	125	3	3,10E+09	158	0	-	-	0
harp2	-6,42E+07	13	5	-7,31E+07	1	1366	-5,59E+07	24	0	-7,26E+07	2	0
liu	4292	187	0	2096	40	246	4100	174	1	4674	212	1
markshare1	234	23300	0	18	1700	680	194	19300	1	230	22900	0
markshare2	553	55200	0	34	3300	2547	365	36400	0	898	89700	0
mas74	16039,8	36	0	13039,8	10	2135	19033,1	61	0	14372,9	22	0
mas76	42875,8	7	0	40121,9	0^+	3320	50124	25	0	**40005,1**	**0**	**0**
misc07	3745	33	0	**2810**	**0**	**1**	3425	22	0	**2810**	**0**	**0**
mkc	-437,49	22	28	-437,49	22	28	-289,95	49	0	-528,53	6	1
mod011	-5,17E+07	5	0	-5,31E+07	3	3132	-4,56E+07	16	1	-4,74E+07	13	0
modglob	3,62E+07	74	0	2,08E+07	0^+	302	2,11E+07	2	0	2,08E+07	0^+	0
momentum1	-	-	3600	-	-	3600	346535	218	223	527461	383	21
net12	337	57	69	337	57	69	337	57	14	-	-	22
nsrand-ipx	57920	13	1	56640	11	10	89120	74	1	55680	9	1
nw04	17514	4	4	**16862**	**0**	**1747**	17856	6	9	16956	1	1
opt1217	**-16**	**0**	**0**	**-16**	**0**	**0**	**-16**	**0**	**0**	**-16**	**0**	**0**
p2756	59846	1816	5	3226	3	79	89266	2757	3	3425	10	0
pk1	31	182	0	**11**	**0**	**3264**	83	655	0	18	64	0
pp08a	12670	72	0	7810	6	2931	10940	49	0	8120	10	0
pp08aCUTS	10350	41	0	7580	3	909	8530	16	0	8100	10	0
protfold	-14	55	40	-20	35	1085	-12	61	268	-20	35	15
qiu	318,8567	340	1	-36,464	73	2572	625,709	571	0	173,979	231	1
rd-rplusc-21	-	-	3600	-	-	3600	171182	3	790	-	-	23
set1ch	84281	55	0	60472,5	11	1516	84167,5	54	0	66772,5	22	0
seymour	438	4	1	428	1	4	445	5	3	434	3	11
sp97ar	7,11E+08	5	11	7,08E+08	5	23	9,41E+08	39	3	6,83E+08	1	13
stp3d	-	-	3600	-	-	3600	-	-	3600	-	-	3016
swath	711,687	52	5	522,37	12	3462	1280,95	174	13	1521,66	226	0
t1717	351388	79	475	286071	46	3449	195779	0	171	340588	74	28
tr12-30	253639	94	3	166120	27	5	163794	25	0	-	-	0
vpm2	17,25	25	0	14,25	4	3509	15,25	11	0	15,25	11	0
l152lav	4770	1	1	**4722**	**0**	**646**	4757	1	0	4760	1	0
stein45	**30**	**0**	**0**	**30**	**0**	**0**	35	17	0	**30**	**0**	**0**
ran8x32	6047	15	0	5383	3	3011	5817	11	0	5837	11	0
ran10x26	5101	19	0	4373	2	3341	4833	13	0	4745	11	0

Continued on next page

Table 2 – continued from previous page

	PTS-first			PTS-3600			OFP			HCPLEX		
Name	Sol. value	Gap %	Time	Sol. value	Gap %	Time	Sol. value	Gap %	Time	Sol. value	Gap %	Time
ran12x21	4597	25	0	3868	6	3378	4231	15	0	4080	11	0
ran13x13	4267	31	0	**3252**	**0**	**1228**	3820	17	0	3508	8	0
binkar10_1	7875,64	17	1	7202,81	7	94	7156,21	6	1	6917,17	3	0
irp	12310,3	1	1	**12159,5**	**0**	**405**	12162,4	0^+	1	12162,4	0^+	0
mas284	94429,6	3	0	**91405,7**	**0**	**166**	99522,7	9	0	93708,1	3	0
prod1	-41	27	0	-51	9	161	-53	5	0	-49	13	0
bc1	3,43703	3	1	3,39703	2	6	5,4391	63	2	3,44084	3	0
bienst1	59,67	28	0	**46,75**	**0**	**9**	55,5	19	0	66,5	42	0
bienst2	70,7	29	0	56,5	3	162	73,6667	35	1	62,1667	14	0
dano3_3	**576,345**	**0**	**84**	**576,345**	**0**	**84**	**576,345**	**0**	**47**	576,396	0^+	69
dano3_4	576,499	0^+	13	**576,435**	**0**	**1599**	**576,435**	**0**	**76**	576,499	0^+	66
dano3_5	578,7	0^+	388	577,1	0^+	439	576,994	0^+	106	578,648	0^+	91
mkc1	-595,49	2	12	-604,91	0^+	2432	-563,1	7	0	-604,86	0^+	1
neos1	24	26	2	**19**	**0**	**5**	68	258	1	21	11	0
neos2	1390,65	206	7	1390,65	206	7	958,977	111	7	-	-	0
neos3	1390,65	277	124	1353,98	267	124	1630,21	342	8	-	-	0
neos4	-4,53E+10	7	1	-4,61E+10	5	3600	-4,81E+10	1	3	-4,83E+10	1	1
neos5	16	7	0	**15**	**0**	**1**	*	*	*	15,5	3	0
neos6	94	13	32	87	5	1643	93	12	12	91	10	4
seymour1	412,052	0^+	2	411,536	0^+	9	427,063	4	3	412,448	0^+	10
swath1	382,308	1	1	**379,071**	**0**	**2306**	439,106	16	2	879,034	132	0
swath2	396,016	3	1	**385,2**	**0**	**2193**	641,544	67	2	1028,07	167	0
acc-0	**0**	-	**1**	**0**	-	**1**	**0**	-	**0**	**0**	-	**0**
acc-1	**0**	-	**3**	**0**	-	**3**	**0**	-	**1**	**0**	-	**3**
acc-2	**0**	-	**57**	**0**	-	**57**	**0**	-	**3**	-	-	8
acc-3	**0**	-	**128**	**0**	-	**128**	**0**	-	**12**	-	-	14
acc-4	-	-	3600	-	-	3600	-	-	3600	-	-	14
acc-5	1	-	649	1	-	649	**0**	-	**2054**	-	-	7
acc-6	**0**	-	**2795**	**0**	-	**2795**	-	-	3600	-	-	9
Mean		1248			93			922			1909	
Mean(57)		1450			96			1083			2003	
Fails		5			5			4			16	

Table 3: Comparison of methods according to the gap range

	PTS-first		PTS-3600		OFP		HCPLEX	
Gap range	Number	%	Number	%	Number	%	Number	%
Less than 1%	14	17.9	32	41.0	16	20.5	19	24.4
1% to 5%	9	11.5	15	19.2	7	9.0	9	11.5
5% to 10%	6	7.7	6	7.7	7	9.0	6	7.7
10% to 25%	10	12.8	5	6.4	14	17.9	15	19.2
25% to 50%	12	15.4	5	6.4	8	10.3	2	2.6
50% to 100%	10	12,8	3	3.8	8	10.3	3	3.8
Greater than 100%	11	14.1	6	7.7	14	17.9	8	10.3
Non-classified	1	1.3	1	1.3	0	0.0	0	0.0
Failures	5	6.4	5	6.4	4	5.1	16	20.5

6 Conclusions

We have conducted a computational study of the core version of parametric tabu search for solving 0–1 MIP problems. In the parametric approach, branching inequalities associated with binary variables, called goal conditions, are imposed indirectly by including them with associated weights in the objective function of the linear programming relaxation of a MIP, denoted (LP′). Parametric tabu search involves measures to classify optimal values of binary variables to (LP′) in suitable sets and assign a response to the variables of each set. Optimal values of binary variables in (LP′) that violate goal conditions define the set G of goal infeasible variables. A measure called goal resistance is used to select the set of variables $x_j, j \in G_p \subseteq G$, which take on new goals in the opposite direction, and the set of variables $x_j, j \in G_s \subseteq G$ that are freed from their goal conditions. The set of potentially goal infeasible variables involve variables with satisfied goals and a much smaller goal resistance compared to the goal infeasible variables. The same responses for goal infeasible variables apply to variables that are potentially goal infeasible. The set D_0 contains binary variables with no goal whose optimal solutions in (LP′) are fractional and have the largest values of a choice preference measure that depends on up and down penalties. Goals are assigned to such variables according to the penalties. Finally, a tabu restriction is associated with an response for a given variable, by forbidding the response from being executed, if the opposing response was executed within the most recent tabu tenure iterations.

We have examined three strategies to measure goal resistance, namely, goal distance (GD), reliability branching (RB) and active-constraint variable ordering (ACVO). The Wilcoxon test shows that ACVO is superior to GD and RB. We have also shown that it is better to have a memory for the weight associated with a goal established in a given iteration that decays exponentially with the number of iteration. In addition, the adaptive strategy that determines the cardinality of the sets G_p, G_s and D_0 according to the conditions of the region that is being visited is more effective than a static strategy.

The Wilcoxon test could not determine any dominance between objective feasibility pump (OFP), CPLEX heuristics (HCPLEX) and parametric tabu search method PTS-first. However, the Wilcoxon test determined that parametric tabu search PTS-3600 outperforms OFP and HCPLEX, which shows the potential of parametric tabu search for obtaining a large number of feasible and high-quality solutions when more running time is allowed. Future research involves the integration of intensification and diversification methods suggested by Glover [17], such as approaches based on exploiting strongly determined and consistent variables, approaches derived from scatter search and methods based on frequency analysis.

Acknowledgments This research was partially funded by the the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). We are also grateful for the valuable comments and suggestions from two anonymous referees.

References

[1] Achterberg, T., Berthold, T. Improving the Feasibility Pump. Discrete Optimization 2007;**4**:77-86.

[2] Achterberg, T., Koch, T., Martin, A. The mixed integer programming library: Miplib, 2003. *http://miplib.zib.de*.

[3] Achterberg, T., Koch, T. , Martin, A. Branching Rules Revisited. Operations Research Letters 2005;**33**:42-54.

[4] Atamtürk, A., Savelsbergh, M. W. P. Integer-programming software systems. Annals of Operations Research 2005;**140**:67-124.

[5] Balas, E., Ceria, S., Dawande, M., Margot, F., Pataki, G. OCTANE: A new heuristic for pure 0–1 programs. Operations Research 2001;**49**(2):207-225.

[6] Balas, E., Martin, C.H. Pivot-and-complement: A heuristic for 0–1 programming. Management Science 1980;**26**(1):86-96.

[7] Balas, E., Schmieta, S., Wallace, C. Pivot and shift-a mixed integer programming heuristic. Discrete Optimization 2004;**1**(1):3-12.

[8] Bénichou, M., Gauthier, J.M., Girodet, P., Hentges, G., Ribière, G., Vincent, O. Experiments in mixed-integer linear programming. Mathematical Programming 1971;**1**:76-94.

[9] Bertacco, L., Fischetti, M., Lodi, A. A feasibility pump heuristic for general mixed-integer problems. Discrete Optimization 2007;**4**:63-76.

[10] Danna, E., Rothberg, E., Le Pape, C. Exploring Relaxation Induced Neighborhoods to Improve MIP Solutions. Mathematical Programming Series A 2005;**102**(1):71-90.

[11] Eckstein, J. Parallel branch-and-bound algorithms for general mixed integer programming on the CM-5. SIAM Journal on Optimization 1994;**4**:794-814.

[12] Faaland, B. H., Hillier, F. S. Interior path methods for heuristic integer programming procedures. Operations Research 1979;**27**(6):1069-1087.

[13] Fischetti, M., Glover, F., Lodi, A. The Feasibility Pump. Mathematical Programming Series A 2005;**104**(1):91-104.

[14] Fischetti, M., Lodi, A. Local Branching. Mathematical Programming Series B 2003;**98**:23-47.

[15] Fischetti, M., Lodi, A. Repairing MIP infeasibility through local branching. Computers & Operations Research 2008;**35**:1436-1445.

[16] Glover, F. Parametric Branch and Bound. OMEGA, The International Journal of Management Science 1978;**6**(2):145-152.

[17] Glover, F. Parametric tabu-search for mixed integer programs. Computers & Operations Research 2006;**33**:2449-2494.

[18] Glover, F. Infeasible/feasible search trajectories and directional rounding in integer programming. Journal of Heuristics 2007;**13**:505-541.

[19] Glover, F., Laguna, M. General purpose heuristics for integer programming -part I. Journal of Heuristics 1997;**2**:343-358.

[20] Glover, F., Laguna, M. General purpose heuristics for integer programming -part II. Journal of Heuristics 1997;**3**:161-179.

[21] Glover, F., Laguna, M. Tabu Search. Kluwer Academic Publisher, Boston, Dordrecht, London, 1997.

[22] Ghosh, S.: DINS, a MIP improvement heuristic, In Proceedings of 12*th* Integer Programming and Combinatorial Optimization – Lecture Notes in Computer Science **4513**, 310–323 (2007).

[23] Golden, B. L., Stewart, W. R. Empirical Analysis of Heuristics in The Traveling Salesman Problem. E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys, eds., John Wiley, 1985:207-250.

[24] Hansen, P. , Mladenović, N., Urošević, D. Variable neighborhood search and local branching. Computers & Operations Research 2006;**33**:3034-3045.

[25] Hillier, F. S. Efficient heuristic procedures for integer linear programming with an interior. Operations Research 1969;**17**:600-637.

[26] Ibaraki, T., Ohashi, T., Mine, H. A heuristic algorithm for mixed-integer programming problems. Mathematical Programming Study 1974;**2**:115-136.

[27] ILOG, Cplex. *http://www.ilog.com/products/cplex*.

[28] Linderoth, J.T., Savelsbergh, M.W.P. A computational study of search strategies for mixed integer programming. Informs Journal on Computing 1999;**11**:173-187.

[29] Løkketangen, A., Glover, F. Solving zero/one mixed integer programming problems using tabu search. European Journal of Operational Research 1998;**106**:624-658.

[30] Løkketangen, A., Jörnsten, K., Storøy, S. Tabu search within a pivot and complement framework. International Transactions in Operational Research 1994;**1**(3):305-316.

[31] Mittelmann, H. Decision tree for optimization software: Benchmarks for optimization software, 2003. *http://plato.asu.edu/bench.html*.

[32] Mosteller, F., Rourke, R. E. K. Sturdy statistics: nonparametrics and order statistics. Addison-Wesley, 1973.

[33] Patel, J., Chinneck, J. W. Active constraint variable ordering for faster feasibility of mixed integer linear programs. Mathematical Programming Series A 2007;**110**:445-474.

[34] Saltzman, R. M., Hillier, F. S. A heuristic ceiling point algorithm for general integer linear programming. Management Science 1992; **38**(2):263-283.