MINISTÉRIO DA CIÊNCIA E TECNOLOGIA

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INPE

RELAXAÇÃO LAGRANGEANA COM DIVISÃO EM *CLUSTERS* PARA ALGUNS PROBLEMAS DE OTIMIZAÇÃO MODELADOS EM GRAFOS DE CONFLITOS

Glaydston Mattos Ribeiro

Proposta de Tese de Doutorado do Curso de Pós-graduação em Computação Aplicada, orientada pelo Dr. Luiz Antonio Nogueira Lorena.

INPE
São José dos Campos
2005

...(...)

RIBEIRO, G. M.

Relaxação lagrangeana com divisão em *clusters* para alguns problemas de otimização modelados em grafos de conflitos / G. M. Ribeiro. – São José dos Campos: INPE, 2005.

..... – (INPE-....-.../...).

1. Relaxação Lagrangeana. 2. Relaxação Lagrangeana*Surrogate*. 3. Geração de Colunas. 4. Branch-and-Price. 5. Particionamento de Grafos.

RESUMO

Muitos problemas de otimização podem ser modelados através de um grafo de conflitos. A presente proposta aborda alguns desses problemas, apresentando uma relaxação lagrangeana (LagClus) obtida após o particionamento do grafo de conflitos em $\overline{P}$ *clusters*. Após o particionamento, as arestas que conectam os *clusters* são relaxadas no sentido lagrangeano e assim, o problema original relaxado pode ser decomposto em $\overline{P}$ subproblemas e resolvido. A vantagem desta relaxação reside no fato de que os subproblemas gerados apresentam estruturas semelhantes ao problema original, conseqüentemente, os limitantes duais obtidos são melhores que uma relaxação lagrangeana sobre todas as arestas do grafo de conflitos. Esta técnica foi aplicada com êxito a problemas como o da rotulação cartográfica de pontos e do carregamento de paletes. Devido aos resultados obtidos, este trabalho vem propor o desenvolvimento de um algoritmo *Branch-and-Price* para esses problemas, assim como a aplicação da LagClus na formulação reduzida, também apresentada na presente proposta, para o problema da programação diária de fotos de um satélite de observação da Terra. Os primeiros resultados obtidos com a técnica de Geração de Colunas são apresentados e já se mostram muito promissores para a realização do algoritmo *Branch-and-Price*. [1]

[1]Este trabalho conta com apoio financeiro do CNPq.

SUMÁRIO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE SÍMBOLOS

V	–	Conjunto de vértices de um grafo
E	–	Conjunto de arestas de um grafo
G	–	Representação de um grafo
V'	–	Subconjunto de vértices do conjunto V
P	–	Problema de otimização
PI	–	Problema de otimização com variáveis inteiras
L_uPI	–	Relaxação do problema PI
DL	–	Problema dual lagrangeano
$\overline{P}$	–	Número de partições de um grafo
$L_\lambda P$	–	Relaxação lagrangeana do problema P
$L_\lambda PMCIV^p$	–	Relaxação do PMCIV associada ao *cluster* p
L_tSP^λ	–	Relaxação lagrangeana/*surrogate* do problema P
D_t	–	Dual da relaxação lagrangeana/*surrogate*
NC	–	Não calculado
A^p	–	Matriz binária de coeficientes das variáveis do *cluster* p pertencentes às arestas entre *clusters*
N	–	Conjunto de vértices com pesos
M	–	Número de restrições entre *clusters*
D^p	–	Matriz binária de coeficientes das restrições de adjacências do *cluster* p
B^n	–	Vetor binário de dimensão n
R^n_+	–	Vetor de número reais não negativos de dimensão n
PC_i	–	Número de posições candidatas de um ponto i
$w_{i,j}$	–	Custo associado à posição candidata $x_{i,j}$
$S_{i,j}$	–	Conjunto de pares de posições candidatas conflitantes com
	–	a posição candidata $x_{i,j}$
C	–	Número de restrições de conflitos ou adjacências
NP	–	Número de pontos a serem rotulados
R^n	–	Vetor de números reais de dimensão n
$CP_{i,j}$	–	Conjunto de todos os pontos que possuem posições candidatas conflitantes com a posição $x_{i,j}$
F	–	Conjunto de fotografias candidatas
m	–	Número possível de pares de elementos foto e câmera
K	–	Conjunto de todas as cliques maximais
CL_k	–	Conjunto de vértices que compõem a clique maximal k
$\hat{N}_i$	–	Conjunto de vértices que possuem uma restrição de adjacência com o vértice i
α	–	Vetor de variáveis duais de comprimento M
β_p	–	Variável dual associada à p-ésima restrição
Z^+	–	Conjunto dos números inteiros não negativos
Z^n	–	Vetor de números inteiros não negativos de comprimento n

X – Conjuntos de números inteiros não negativos
Y – Conjuntos de números inteiros não negativos
a – Função que descreve as restrições de sobreposição no palete
L – Comprimento do palete
W – Largura do palete
l – Comprimento da caixa
w – Largura da caixa

LISTA DE SIGLAS E ABREVIATURAS

LagClus	–	Relaxação Lagrangeana com *Clusters*
PMCIV	–	Problema de Máximo Conjunto Independente de Vértices
PPG	–	Problema de Particionamento de Grafo
PMCIVP	–	Problema de Máximo Conjunto Independente de Vértices com Pesos
PRCP	–	Problema da Rotulação Cartográfica de Pontos
PMNRSC	–	Problema do Máximo Número de Rótulos Sem Conflitos
PMNC	–	Problema da Minimização do Número de Conflitos
AG	–	Algoritmo Genético
ACS	–	*Ant Colony System*
AGC	–	Algoritmo Genético Construtivo
LagSur	–	Relaxação Lagrangeana/*Surrogate*
PMR	–	Problema Mestre Restrito
B&P	–	*Branch-and-Price*
PCP	–	Problema do Carregamento de Paletes
RLF	–	*Recursive-Largest-First*
RSF	–	*Recursive-Smallest-First*
PPDF	–	Problema da Programação Diária de Fotos

CAPÍTULO 1

INTRODUÇÃO

Muitas estruturas envolvendo situações reais podem ser convenientemente representadas utilizando grafos. Um grafo G é representado através de um conjunto V de vértices e um conjunto E de pares de vértices ou simplesmente arestas, sendo o grafo então representado por $G = (V, E)$. Por exemplo, os vértices em um grafo poderiam representar diferentes cidades de um país, e as linhas que as conectam poderiam indicar rodovias ou simplesmente estradas.

Existe na literatura um tipo de grafo denominado grafo de conflitos que aparece freqüentemente em muitos problemas. Um grafo de conflitos representa relações lógicas entre variáveis binárias (Atamtürk et al., 2000). Mais precisamente, um grafo de conflitos tem um vértice para cada variável binária, e uma aresta entre dois vértices quando somente um deles pode receber o valor um na solução do problema. Essas arestas são comumente conhecidas como arestas de adjacências ou conflitos.

O otimização em grafos de conflitos aparece em vários problemas como em modelos de alocação (Gerrard e Church, 1996), *Anti-Covering* (Murray e Church, 1997b), carregamento de paletes (Dowsland, 1987; Alvarez-Valdes et al., 2004), planejamento de florestas e de colheitas (Churh et al., 1998; Goycoolea et al., 2005), redes de telefonia celular (Björkund et al., 2005) e rotulação cartográfica (Ribeiro e Lorena, 2004c; Yamamoto e Lorena, 2005).

Entre os problemas que mais utilizam grafos de conflitos está o Problema do Máximo Conjunto Independente de Vértices (PMCIV), que tem como objetivo obter o maior subconjunto de vértices tal que todos eles sejam independente entre si, ou seja, não apresente nenhuma aresta de conflito. Utilizando a teoria dos grafos, esse problema pode ser formalmente definido como: obter a cardinalidade do maior subgrafo induzido de G que seja totalmente desconexo (Harary, 1972).

Entretanto, em muitos problemas como os citados acima, o número de arestas (restrições de conflitos ou adjacências) presentes no grafo G é muito grande, inviabilizando assim, o processo de obtenção da solução ótima, o que justifica a classificação desse problema na literatura como *NP-Hard* (Harary, 1972). Com isso, existe na literatura um predomínio de algoritmos heurísticos para resolver tais problemas (ver Dowsland (1987); Klotz (2002)), assim como o uso de metaheurísticas conhecidas como Busca Tabu (Pureza e Morabito, 2005; Yamamoto et al., 2002), *Simulated Annealing* (Björkund et al., 2005), Algoritmos

Genéticos (Verner et al., 1997) e Algoritmo Genético Construtivo (Yamamoto e Lorena, 2005).

Devido à dificuldade inerente de tais problemas, muitos pesquisadores buscam reduzir o número de restrições de conflitos, utilizando o conceito de cliques. Dentre os trabalhos relacionados estão Murray e Church (1996, 1997a,b) que conseguem reduzir de forma significativa o número de restrições geradas. Recentemente Goycoolea et al. (2005), estudando o problema de planejamento florestal, apresentou também contribuições no sentido de reduzir tais restrições.

Mesmo com esses métodos, muitos *softwares* bem conhecidos de otimização não conseguem resolver de forma exata vários problemas, com isso, muitos pesquisadores recorrem às relaxações para obter bons limitantes. Dentre as mais citadas estão as relaxações Lagrangeana e Lagrangeana/*Surrogate*(Narciso e Lorena, 1999). Entretanto, problemas relaxados podem também apresentar dificuldades para serem resolvidos (ver Murray e Church (1997b); Letchford e Amaral (2001)). Por outro lado, diferentemente de uma heurística ou metaheurística, uma relaxação pode apresentar uma informação dual de boa qualidade, o que permite avaliar o quão próximo, a melhor solução encontrada para o problema, está da solução ótima.

Existem outras duas questões importantes e que geralmente estão presentes nos grafos de conflitos. A primeira diz respeito a regiões agrupadas que formam aglomerados (*clusters*) de vértices e arestas. A segunda está relacionada a classificação do grafo como esparso que está indiretamente ligada à questão dos *clusters*.

A Figura 1.1 exemplifica a questão dos *clusters*. Na Figura 1.1(a) têm-se dois agrupamentos bem definidos, em (b) pode-se observar separadamente as arestas que conectam estes *clusters* que, se removidas, formam dois subgrafos com as mesmas características do grafo original (ver Figura 1.1(c)).

Entretanto, se os dois subproblemas, referentes aos subgrafos, forem independentemente resolvidos, não há garantias de se estar resolvendo o problema original. Por outro lado, se as arestas (restrições de conflitos) removidas mostradas na Figura 1.1(b) forem relaxadas no sentido lagrangeano, estas passam a ser consideradas e com isso, pode-se obter um limitante de qualidade para o problema original.

A idéia acima envolve um outro problema, o particionamento de grafos, que também é classificado como *NP-Hard*(Garey et al., 1976). No entanto, existem várias heurísticas eficientes que podem particionar grafos em $\overline{P}$-partes em questão de segundos, com

particionamentos de boa qualidade (ver Karypis e Kumar (1998a)).

Assim, o particionamento de grafos associado à relaxação lagrangeana, pode ser uma boa estratégia para fornecer limitantes bons para problemas modelados em grafos de conflitos, sendo este o enfoque desta proposta.

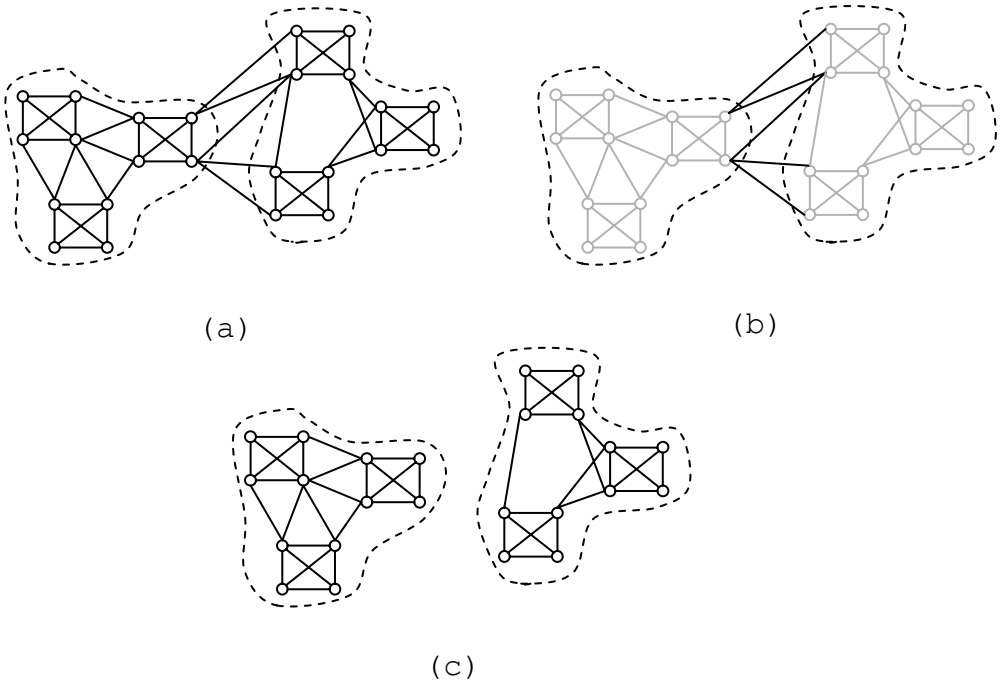


FIGURA 1.1 – Exempo de PMCIV. (a) Grafo de conflito, (b) arestas de ligação (c) *clusters* ou subproblemas obtidos.

1.1 Objetivos e organização do trabalho

Após as exposições mostradas anteriormente, esta proposta tem por objetivo apresentar a relaxação lagrangeana com *clusters* (LagClus) e seus resultados para os problemas de rotulação cartográfica e carregamento de paletes, assim como, propor o desenvolvimento de um algoritmo *Branch-and-Price*, baseado na idéia da LagClus, para problemas que podem ser modelados em grafos de conflitos.

Inicialmente, no Capítulo 2 será feita uma análise sobre otimização combinatória, relaxação lagrangeana e particionamento de grafos. Em seguida, no Capítulo 3 é apresentada a modelagem clássica do problema de máximo conjunto independente de vértices assim como a LagClus. No Capítulo 4 é apresentada a aplicação da LagClus ao problema da rotulação cartográfica de pontos. No Capítulo 5 é apresentada uma formulação reduzida obtida para o problema da programação de fotos de um satélite de observação. Em seguida, no Capítulo 6 é apresentado o problema do carregamento de

paletes e os resultados obtidos com a LagClus. Este capítulo também apresenta o processo de geração de colunas, fase inicial do *Branch-and-Price*, e seus resultados também para o problema do carregamento de paletes. No Capítulo 7 serão apresentadas as próximas atividades e o cronograma de trabalho proposto. Por último, ainda no Capítulo 7, são descritas as principais conclusões e contribuições já presentes nesta proposta.

CAPÍTULO 2

CONCEITOS DE RELAXAÇÃO LAGRANGEANA E PARTICIONAMENTO DE GRAFOS

2.1 Otimização combinatória

Problemas de otimização podem ser divididos em três categorias: aqueles que trabalham com variáveis contínuas, os que trabalham com variáveis discretas, e os mistos. Em problemas contínuos, geralmente procura-se por um conjunto de números reais. Já em problemas discretos ou combinatoriais, procura-se por um objeto de um conjunto finito, tipicamente um inteiro, conjunto, permutação ou grafo (Papadimitriou e Steiglitz, 1998).

Dado um conjunto finito $N = \{1, ..., n\}$ com pesos c_j para todo $j = \{1, ..., n\}$, e um conjunto F de subconjuntos factíveis de N, um problema de otimização combinatória P pode ser expresso como:

$$v(P) \quad = Min(ou \quad Max)\{f(s)\} \tag{2.1}$$

Sujeito a:

$$s \in F \tag{2.2}$$

Sendo $f(s)$ uma função que representa o peso da combinação s tal que $s \in F$.

O processo para se determinar a melhor combinação factível é o foco da otimização combinatória, entretanto isto pode consumir vários recursos computacionais tornando-se assim, fruto de pesquisa intensa.

Existem na literatura, várias classes de problemas de otimização combinatória, largamente estudados, mas que ainda hoje apresentam-se desafiadoras, como é o caso dos problemas de alocação, cobertura de conjuntos e do caixeiro viajante. Todos apresentam como solução ótima, como mostrado na Restrição 2.2, algum subconjunto de um conjunto finito. Com isso, teoricamente, esses problemas poderiam ser resolvidos por enumeração. No entanto, como mostrado em Parker e Rardin (1988) e Papadimitriou e Steiglitz (1998), a enumeração completa só é viável para problemas pequenos, tornando-se impraticável para problemas de grande porte. Com isso, recorre-se a outros métodos como heurísticas, metaheurísticas e relaxações.

2.2 Relaxação lagrangeana

Considere o problema (P) apresentado em 2.1-2.2 como um problema de maximização. Para garantir que uma solução x^* é ótima, basta encontrar um limitante superior $\overline{x}$ de tal forma que $v(x^*) = v(\overline{x})$.

Isso é importante pois, se um algoritmo é capaz de encontrar uma ordem decrescente de limitantes duais $v(\overline{x_1}) > v(\overline{x_2}) > ... > v(\overline{x_s}) \geq v(x^*)$, e uma ordem crescente de limitantes primais $v(\underline{x_1}) < v(\underline{x_2}) < ... < v(\underline{x_i}) \leq v(x^*)$, e parar quando $v(\overline{x_s}) - v(\underline{x_i}) \leq \varepsilon$; dependendo do valor de ε, é possível garantir que uma solução é ótima, nesse caso $\underline{x_i}$. O processo descrito aqui, pode ser melhor representado graficamente como mostra a Figura 2.1.

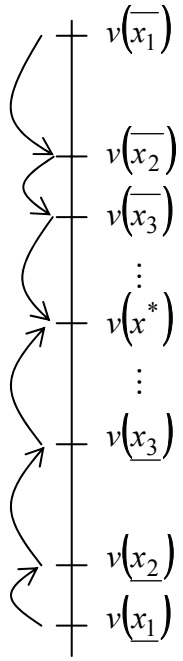


FIGURA 2.1 – Evolução dos limitantes duais e primais (Wolsey, 1998).

Entretanto, encontrar limitantes duais de boa qualidade não é uma tarefa simples. Um dos métodos mais utilizados é a relaxação que pode trabalhar ampliando o espaço de busca, alterando assim, o conjunto de soluções factíveis; ou substituindo a função objetivo por uma função que tem um valor maior ou igual ao da função objetivo original.

Dentro desse contexto, está a relaxação lagrangeana que tem provado sua eficiência (ver Held e Karp (1971); Beasley (1990); Murray e Church (1997b); Lorena e Senne (1999); van Krieken et al. (2004)).

As definições que serão apresentadas a seguir, baseiam-se nos trabalhos de Parker e Rardin (1988); Narciso (1998); Wolsey (1998).

Suponha o seguinte problema inteiro *(PI)* $Max\{cx : Ax \leq b, x \in X \subseteq Z^n\}$, onde c e b são vetores, A é uma matriz e x um vetor de variáveis inteiras. Dado que o problema é de difícil solução, uma maneira de resolvê-lo é aplicando a relaxação lagrangeana sobre a restrição $Ax \leq b$, assim temos a seguinte proposição:

Proposição 2.2.1 *Seja* (L_uPI) *a relaxação lagrangeana dada por* $Max\{cx + u(b - Ax) : x \in X\}$. *Então* $v(L_uPI) \geq v(PI)$ *para todo* $u >= 0$.

Porém, a relaxação lagrangeana (L_uPI) é mais interessante se o valor da mesma se aproximar da solução ótima de (PI). Para isso, é importante a escolha do multiplicador lagrangeano u, que pode ser obtido resolvendo o seguinte problema dual: (DL) $Min\{(L_uPI) : u \geq 0\}$.

Passo 0: ***Inicialização.*** Faça $u^1 \geq 0$, $k = 1$, $lb = -\infty$ e $ub = +\infty$.

Passo 1: ***Relaxação Lagrangeana.***
Resolver (L_uIP) obtendo x^k e $v(L_uIP)$.
Construir uma solução factível x^f para (IP) e obter o respectivo v_f.

Passo 2: ***Guardar melhor solução e melhor limitante***
Atualizar $lb = Max\{lb, v_f\}$.
Atualizar $ub = Min\{ub, v(L_uIP)\}$.

Passo 3: ***Calcular passo do subgradiente.***
Calcular $g_j^k = b_j - Ax_j^k$ para todo $j = 1,...,n$.
Atualizar o passo $\theta = \pi_k(ub - lb)/||g^k||^2$.
Calcular os novos multiplicadores $u_j^{k+1} = Max\{0, u_j^k + \theta g_j^k\}$ para todo $j = 1,...,n$.
$k = k + 1$.

Passo 4: ***Teste de parada.***
Se atingir alguma condição de parada, parar o algoritmo de subgradientes, caso contrário, voltar ao passo 1.

FIGURA 2.2 – Algoritmo de subgradientes.

Teorema 2.2.2 *Sejam* (PI), (L_uPI) *e* (DL) *como definidos acima. Então para* $u >= 0$, $v(L_uPI) >= v(PI)$ *e, conseqüentemente,* $v(DL) >= v(PI)$.

Teorema 2.2.3 *Sejam* (PI), (L_uPI) *e* (DL) *como definidos acima. Se* x *é a solução*

de (L_uPI) *para algum valor de* $u >= 0$*, e se* $Ax <= b$ *e* $u(b - Ax) = 0$*, então* x *é a solução ótima de* (PI).

Os Teoremas 2.2.2 e 2.2.3 mostram que com a relaxação lagrangeana é possível obter a solução ótima de um problema como foi previsto pela Proposição 2.2.1. Para isso, existe a necessidade de se escolher os multiplicadores $u >= 0$. No entanto, resolver o dual (DL) não é uma tarefa simples, pois para cada valor de u deve-se resolver uma relaxação lagrangeana (L_uPI). Entretanto, (L_uPI) em função do multiplicador u, apresenta a propriedade de ser linear por partes. Por outro lado, dado que a função é linear por partes, não é possível garantir que ela seja sempre diferenciável.

Com isso, utiliza-se o conceito de subgradientes que aproximam, no sentido Euclidiano, as soluções de um problema relaxado à solução ótima (Parker e Rardin, 1988). A Figura 2.2 descreve os principais passos para o algoritmo de subgradientes.

2.3 Problema de particionamento de grafos

Muitos problemas combinatoriais podem ser descritos como o problema de encontrar um ou mais conjuntos de vértices de um dado grafo, minimizando uma função objetivo que está relacionada às arestas que conectam os subconjuntos. Tais problemas incluem, por exemplo, armazenamento eficiente em grandes banco de dados (Shekhar e Liu, 1996) e mineração de dados (Mobasher et al., 1996; Karypis e Kumar, 1999).

Genericamente, esse problema pode ser expresso como: dado um grafo $G = (V, E)$ (onde V é um conjunto de vértices com pesos e E é um conjunto de arestas também com pesos) e um número $\overline{P}$, encontre $\overline{P}$ subconjuntos $V_1, V_2, ..., V_{\overline{P}}$ de V tal que:

a) $\bigcup_{i=1}^{\overline{P}} V_i = V$ e $V_i \cap V_j = \emptyset$ para todo $i \neq j$;

b) $W(i) \approx W/\overline{P}$, $\forall i = 1, 2, ..., \overline{P}$ onde $W(i)$ e W representam, respectivamente, as somas dos pesos dos vértices pertencentes a V_i e V;

c) O "tamanho do corte", isto é, a soma dos pesos das arestas que conectam os subconjuntos V_i $\left(i = 1, 2, ..., \overline{P}\right)$ deve ser mínima.

Qualquer conjunto $\left\{V_i \subseteq V : 1 \leq i \leq \overline{P}\right\}$ é chamado de uma partição se satisfaz a condição (a), ou seja, cada V_i é então uma parte de um $\overline{P}$-particionamento. Uma bissecção é um *2*-particionamento. Um particionamento que satisfaz a condição (b) é um particionamento balanceado.

O problema do particionamento de grafos (PPG) é classificado como *NP-Hard* (Garey et al., 1976) assim, ainda não existe um algoritmo função do número de vértices ou

de arestas, que encontre um particionamento ótimo em tempo polinomial. Algoritmos que podem garantir uma solução ótima (ver por exemplo o algoritmo de Karisch et al. (1997)) são indicados para grafos pequenos, não sendo aplicáveis em problemas de grande porte. Entretanto, existem várias heurísticas que diferem em relação a qualidade do particionamento, porém são rápidas.

Existem grafos que são particionados mais de uma vez devido às características do problema. Quando em um particionamento, tanto V quanto E não se alteram, diz-se que se trata de um caso estático, resolvido muitas vezes com algoritmos conhecidos como seqüenciais. No entanto, em alguns casos o grafo se altera de forma incremental, por exemplo em métodos de elementos finitos adaptativos os vértices e arestas podem ser removidos ou adicionados durante os cálculos (Fjällström, 1998). Nesses casos, tem-se um caso dinâmico, e o particionamento de grafos é feito repetidamente. Com isso, os algoritmos mais interessantes são os que fazem o particionamento paralelo, aproveitando informações dos particionamentos anteriores.

2.3.1 Algoritmos seqüenciais

Pesquisadores têm desenvolvido muitos algoritmos seqüenciais para o PPG. Estes baseiam-se em métodos de melhoramentos locais que primeiramente bisseccionam o grafo, e tentam decrescer o tamanho do corte, usando algum método de busca local. Esses métodos são aplicados de forma recursiva para obter um $\overline{P}$-particionamento. Os algoritmos mais citados na literatura são o de Kernighan e Lin (1970) nomeado de algoritmo KL, o *Simulated Annealing* de Johnson et al. (1989), a Busca Tabu de Rolland et al. (1996), e o Algoritmo Genético de Bui e Moon (1996).

2.3.2 Métodos globais

Um método global toma como entrada um grafo G e um inteiro $\overline{P}$, e gera uma $\overline{P}$-partição. Muitos desses métodos são recursivos, ou seja, eles bisseccionam o grafo e em cada subgrafo gerado, é repetido este processo até encontrar o número $\overline{P}$ de partições. No entanto, esses métodos são aperfeiçoados com o uso de métodos de melhoramentos locais.

Os métodos classificados como globais ainda apresentam variações que dependem do modo como atuam no particionamento. Muitos grafos apresentam uma localização geométrica (coordenadas), assim, muitos algoritmos utilizam essa informação para realizar o particionamento, sendo estes classificados como algoritmos de particionamento geométricos (Miller et al., 1993). Outros grafos não apresentam essa propriedade e os métodos de particionamentos são classificados como métodos sem coordenação, levando

então em consideração, a estrutura combinatorial do grafo. Dentre estes, os mais importantes são os métodos que utilizam em seu particionamento procedimentos para obtenção do maior subconjunto independente de vértices em cada subgrafo gerado.

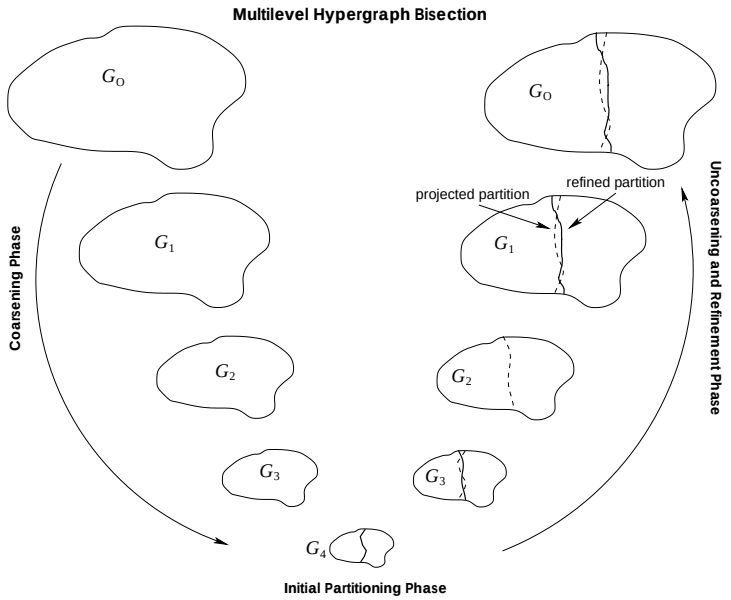


FIGURA 2.3 – Bissecção usando multi-nível (Karypis, 2002).

Um dos trabalhos mais citados na literatura é o da bissecção recursiva de Simon (1991). Esse método inicia localizando dois vértices que possuem o maior caminho mínimo entre si, e em seguida um deles é selecionado. Usando procura em largura, a distância desse vértice em relação aos demais é determinada. Finalmente, os vértices são ordenados em relação a essa distância, e o conjunto ordenado é dividido em dois subconjuntos de mesmo tamanho, e assim sucessivamente.

Outra técnica bastante empregada é a bissecção recursiva em multi-níveis (Karypis e Kumar, 1998a,c; Karypis, 2002). Ela primeiramente vai agrupando os vértices do grafo, formando grafos menores, e em seguida, vai desagrupando e realizando uma bissecção recursiva do grafo, passando por uma fase de refinamento. A Figura 2.3 apresenta de forma esquemática esse processo. Durante a fase de desagrupamento, as linhas pontilhadas indicam os particionamentos projetados e as linhas sólidas, os particionamentos que foram produzidos após o refinamento.

2.3.3 Algoritmos paralelos

Muitos dos algoritmos apresentados nesta seção são formulações paralelas dos métodos apresentados nas seções anteriores. Alguns desses métodos, em particular os métodos

geométricos, são relativamente fáceis de se paralelizar (Fjällström, 1998).

A idéia por trás desses algoritmos é dividir a tarefa de particionamento entre diversos processadores, e com isso, ganhar em tempo computacional, permitindo trabalhar com grafos maiores.

Dentre os trabalhos mais citados estão o de Gilbert e Zmijewski (1987) que é baseado no algoritmo de Kernighan e Lin (1970), JOSTLE-D (Walshaw et al., 1997; Walshaw, 2004) e o algoritmo de reparticionamento paralelo e seqüencial de Schloegel et al. (1997).

2.3.4 Softwares disponíveis

Existem alguns softwares na literatura disponíveis para particionamento de grafos, sendo eles:

- CHACO (Hendrickson e Leland, 1995). Contém implementação de várias técnicas de particionamento, inclusive a heurística KL;
- METIS (Karypis e Kumar, 1998a,c,b). Eles também têm trabalhado com o PARMETIS (Karypis e Kumar, 1997) que é baseado em algoritmos de particionamentos paralelos;
- MESH PARTITIONING TOOLBOX (Gilbert et al., 1995). Inclui uma implementação em Matlab de seu algoritmo;
- JOSTLE (Walshaw et al., 1997);
- TOP/DOMDEC (Simon e Farhat, 1993). Inclui implementação de um algoritmo guloso, assim como outros.

Maiores detalhes sobre os softwares, assim como pesquisadores, problemas testes, paralelismo em grafo e teoria podem ser obtidos no seguinte site <http://rtm.science.unitn.it/intertools/graph-partitioning/links.html>.

2.4 Considerações finais

Este capítulo procurou mostrar de forma resumida, os conceitos presentes na relaxação lagrangeana e os algoritmos para o problema de particionamento de grafos. Observou-se que a relaxação lagrangeana é uma boa opção para obter bons limites duais e isso permite avaliar a solução de um problema. Por outro lado, o particionamento de grafo não é um problema simples, e como pôde ser visto, existem diversas técnicas usadas no processo de solução de um problema de particionamento.

As duas técnicas apresentadas neste capítulo, constituem a base da relaxação lagrangeana com *clusters* que será apresentada no próximo capítulo e exemplificada com um dos problemas objetos desta proposta, que é o problema do máximo conjunto independente de vértices.

CAPÍTULO 3

RELAXAÇÃO LAGRANGEANA COM CLUSTERS

A relaxação lagrangeana com formação de *clusters*, conforme mencionado no Capítulo 1, explora o fato de que várias aplicações podem ser modeladas através de um grafo de conflitos, e em alguns casos, esses são bem adaptados para uma fase de particionamento.

A idéia da LagClus surgiu do trabalho de Hicks et al. (2004) no qual os autores desenvolveram um algoritmo *Branch–and–Price* para o Problema de Máximo Conjunto Independente de Vértices com Pesos (PMCIVP), que é semelhante ao PMCIV porém neste caso, os vértices apresentam pesos. Em seu trabalho, os autores particionam o grafo de conflitos, obtendo subgrafos (subproblemas) induzidos menores que são facilmente resolvidos. O problema original é então formulado usando a decomposição Dantzig-Wolfe (Bazaraa et al., 1990) e esses subproblemas passam a gerar colunas para a decomposição, resolvendo o problema original. Os autores foram bem sucedidos nos testes realizados, entretanto os tempos computacionais foram altos em alguns casos. Isso conduziu ao desenvolvimento da LagClus para conseguir, através da relaxação lagrangeana, resultados tão bons quanto os obtidos por Hicks et al. (2004), mas em um tempo computacional menor.

Na Seção 3.1 é apresentado matematicamente o PMCIV, em seguida, na Seção 3.2, é mostrado o processo de aplicação da LagClus, assim como sua formulação para o PMCIV. Na Seção 3.3 é apresentado um exemplo da aplicação da LagClus e por último, são feitas algumas considerações sobre a LagClus na Seção 3.4.

3.1 Máximo conjunto independente de vértices

O Problema do Máximo Conjunto Independente de Vértices (PMCIV) é um problema clássico, bastante estudado na literatura. O PMCIV normalmente está embutido na aplicação e com isso, surge em vários campos como em teoria da codificação, visão computacional e estudos químicos (ver Bomze et al. (1999)).

Devido a sua grande área de aplicação, existem várias técnicas de solução propostas na literatura. Técnicas exatas incluem enumeração explicita de conjuntos independentes de vértices (Bron e Kerbosch, 1973), *Branch-and-Bound* (Balas e Xue, 1991, 1996; Balas e Yu, 1986; Bourjolly et al., 1997; Carraghan e Pardalos, 1990; Östergard, 2002; Wood, 1997), e formulações contínuas com *Branch-and-Bound* (Barnes, 2000; Mehrotra e Trick, 1996). Além disso, várias heurísticas foram propostas como os algoritmos de contração de vértices (Brigham e Dutton, 1981; Leighton, 1979; Hertz, 1990), e a heurística gulosa

de Kopf e Ruhe (1987). Existem ainda heurísticas de busca local que procuram melhorar uma determinada solução dada, por exemplo, por uma heurística gulosa. Entre os trabalhos relacionados estão os de Feo et al. (1994) e Alon et al. (1986).

Por outro lado, várias metaheurísticas também já foram utilizadas para resolver o PMCIV. Aarts e Korst (1989) propuseram o uso da metaheurística *Simulated Annealing*, Godbeer et al. (1988) usaram Redes Neurais, Bui e Eppley (1995) utilizaram Algoritmos Genéticos, e Soriano e Gendreau (1996); Gendreau et al. (1999) Busca Tabu.

O PMCIV pode ser modelado como segue. Considere $G = (V, E)$ um grafo tal que V representa um conjunto de vértices x_v e E um conjunto de arestas (u, v) sendo $u, v \in V$ e $u \neq v$. Considere ainda que não existem pesos relacionados aos vértices ou arestas. Assim, o problema de máximo conjunto independente de vértices consiste em obter um subconjunto $V' \subseteq V$ tal que todo par de vértices de V' não é adjacente, isto é, se $r, w \in V'$, então $(r, w) \notin E$. Com isso, o PMCIV pode ser modelado da seguinte maneira:

$$v\,(PMCIV) \quad = \quad Max \left\{ \sum_{v \in V} x_v \right\} \tag{3.1}$$

Sujeito a:

$$x_u + x_v \leq 1 \qquad \forall\,(u, v) \in E \tag{3.2}$$

$$x_v \in \{0, 1\}\, \forall v \in V \tag{3.3}$$

Se $x_v = 1$ o vértice v está incluído no conjunto independente, caso contrário $x_v = 0$. As restrições definidas pela Equação 3.2 garantem que dois vértices adjacentes não podem estar presentes simultaneamente na solução do problema. Já a Equação 3.3 indica que todas as variáveis x_v são binárias.

A Figura 3.1 (a) apresenta um grafo. Em (b) tem-se a modelagem matemática do PMCIV para esse grafo, e em (c), a solução ótima é representada pelos vértices negros.

O conjunto de restrições definido na Equação 3.2, como dito no Capítulo 1, varia conforme o número de arestas podendo inviabilizar o processo de solução. Entretanto, o problema pode ser resolvido aplicando-se uma relaxação lagrangeana sobre esse conjunto de restrições. Porém, como apontam Letchford e Amaral (2001) ao trabalharem com problemas de carregamento de paletes, em alguns casos essa relaxação não apresenta bons resultados, podendo até ser considerada como "fraca", necessitando assim de algo mais para torná-la eficiente. Normalmente, isso é feito considerando uma outra característica

do problema, que é modelada como uma restrição e adicionada ao problema relaxado.

A LagClus entretanto, pode ser considerada como uma relaxação "forte", pois somente parte do conjunto de restrições definidos na Equação 3.2 é relaxado. E ainda sim, o número de restrições relaxadas varia de acordo com o número de particionamentos (*clusters*) obtidos.

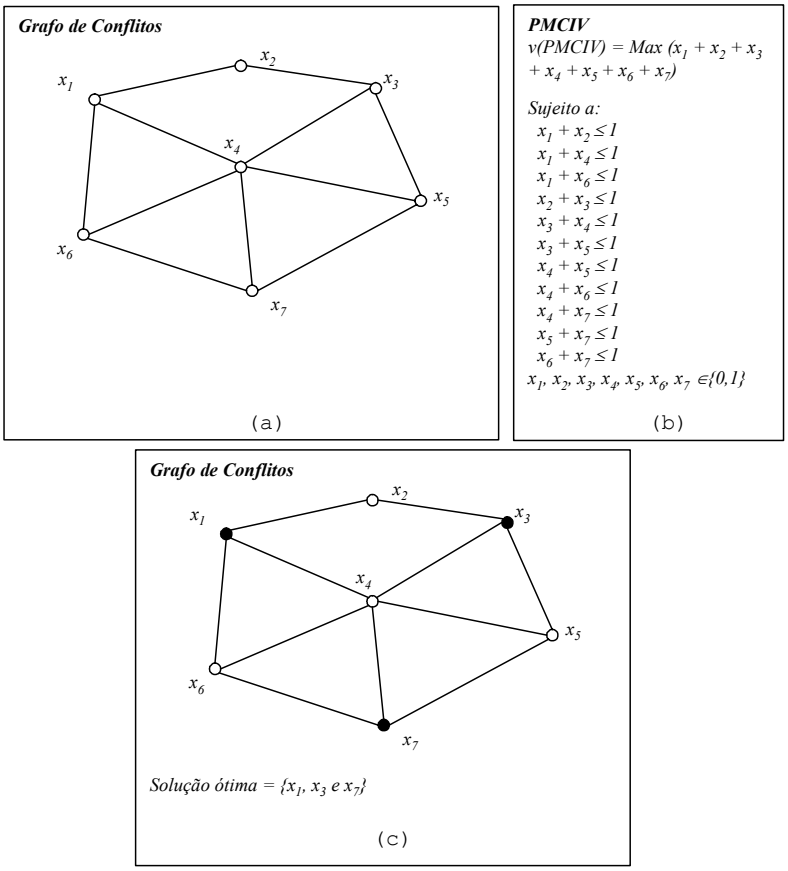


FIGURA 3.1 – PMCIV. (a) Grafo de conflito, (b) formulação e (c) solução ótima.

3.2 Processo de aplicação da LagClus

A relaxação lagrangeana com clusters é aplicada da seguinte maneira:

a) Aplicar uma heurística de particionamento de grafos para dividir o grafo G em $\overline{P}$ *clusters* de mesma cardinalidade, obtendo um $\overline{P}$-particionamento. O PMCIV pode ser agora representado através da função objetivo definida na Equação 3.1 sujeita a 3.2-3.3, sendo agora as restrições de adjacência 3.2 divididas em dois conjuntos: um com restrições de adjacências intra *clusters* e outro com restrições de adjacências entre *clusters* ou arestas de conexão;

b) Um vetor de multiplicadores λ relaxam, no sentido lagrangeano, as restrições de adjacências entre *clusters*;

c) A relaxação lagrangeana resultante é decomposta em $\overline{P}$ subproblemas e resolvida.

É interessante observar que os subproblemas obtidos têm as mesmas características do problema original, porém em escala menor, e podem ser resolvidos de forma exata usando algum *solver* comercial.

Aplicada a primeira etapa da LagClus, o PMCIV pode ser formulado como:

$$v\,(PMCIV) = Max \sum_{p=1}^{\overline{P}} x^p \tag{3.4}$$

Sujeito a:

$$+A^1x^1 \quad +A^2x^2 \quad \cdots \quad +A^{\overline{P}}x^{\overline{P}} \quad \leq \quad 1 \tag{3.5}$$

$$\begin{array}{rrrrcc} +D^1x^1 & & & & \leq & 1 \\ & +D^2x^2 & & & \leq & 1 \\ & & \cdots & & \leq & \vdots \\ & & & +D^{\overline{P}}x^{\overline{P}} & \leq & 1 \end{array} \tag{3.6}$$

$$x^1 \in B^{n_1} \quad \cdots \quad \cdots \quad x^{\overline{P}} \in B^{n_{\overline{P}}} \tag{3.7}$$

Sendo:

- A^p uma matriz binária de dimensão M x $|V|$ que representa os coeficientes das variáveis x^p do *cluster* p presentes nas M restrições de adjacências entre *clusters*;

- D^p uma matriz binária de dimensão $|E| - M$ x $|V|$ que representa os coeficientes das variáveis x^p do *cluster* p presentes nas restrições de adjacências intra *clusters*;

- B^{n_p} um vetor de variáveis binárias do *cluster* p de comprimento n_p.

Como pode-se observar, a restrição 3.5 se relaxada, permite dividir o problema obtido em $\overline{P}$ subproblemas distintos. Então, aplicando a relaxação lagrangeana sobre esse conjunto de restrições, sendo $\lambda \in R_+^M$ tal que M é o número de restrições presentes em 3.5, ou seja o número de arestas particionadas, obtêm-se o seguinte problema relaxado:

$$v\left(L_\lambda PMCIV\right) = Max\left\{\sum_{p=1}^{\overline{P}}\left(1 - \lambda A^p\right)x^p + \sum_{m=1}^{M}\lambda_m : x^p \in X^p\right\} \tag{3.8}$$

Sendo:

$$X^p = \{x^p \in B^{n_p} : D^p x^p \leq 1\} \tag{3.9}$$

Agora reescrevendo o problema acima em função de $\overline{P}$ obtém-se:

$$v\left(L_\lambda PMCIV\right) = \sum_{p=1}^{\overline{P}} Max\left\{\left(1 - \lambda A^p\right)x^p : x^p \in X^p\right\} + \sum_{m=1}^{M}\lambda_m \tag{3.10}$$

Então, o problema relaxado acima pode ser facilmente separável em $\overline{P}$ subproblemas e resolvido.

3.3 Exemplo de aplicação da LagClus

Considere a Figura 3.2. O problema representado pelo grafo mostrado em (a) pode ser modelado como:

$$v(PMCIV) = Max\left\{x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7\right\} \tag{3.11}$$

Sujeito a:

$$
\begin{aligned}
& x_1 + x_2 \leq 1 \\
& x_1 + x_4 \leq 1 \\
& x_1 + x_6 \leq 1 \\
& x_2 + x_3 \leq 1 \\
& x_3 + x_4 \leq 1 \\
& x_3 + x_5 \leq 1 \\
& x_4 + x_5 \leq 1 \\
& x_4 + x_6 \leq 1 \\
& x_4 + x_7 \leq 1 \\
& x_5 + x_7 \leq 1 \\
& x_6 + x_7 \leq 1 \\
& x_1, x_2, x_3, x_4, x_5, x_6, x_7 \in \{0, 1\}
\end{aligned}
\tag{3.12}
$$

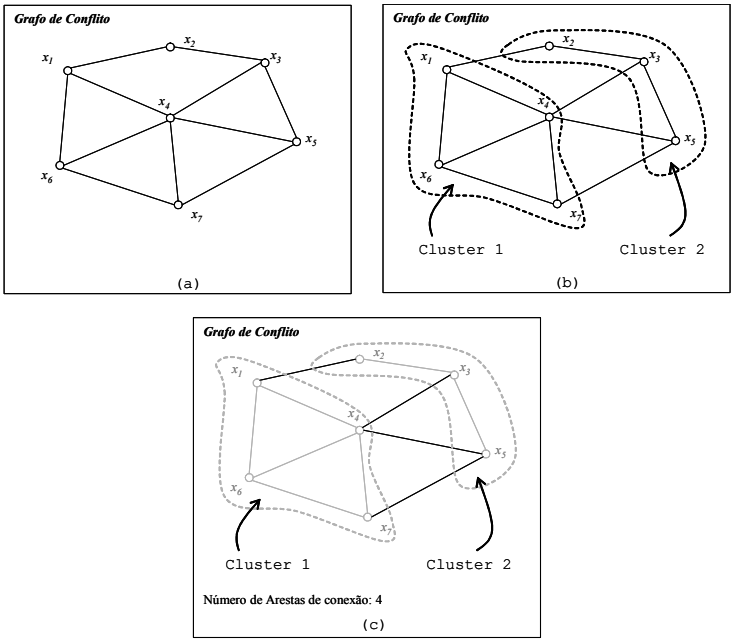


FIGURA 3.2 – Exemplo de aplicação da LagClus. (a) Grafo, (b) particionamento e (c) arestas de conexão.

Suponha uma partição do tipo *2*-particionamento como mostrado na Figura 3.2(b), assim, $\overline{P} = 2$. As arestas mostradas na Figura 3.2(c) representam as arestas que devem ser relaxadas pois são estas as que conectam os *clusters*.

Com isso, esse problema poder ser reescrito como:

	Cluster 1				*Cluster* 2		
$v(PMCIV) =$	$+x_1$	$+x_4$	$+x_6$	$+x_7$	$+x_2$	$+x_3$	$+x_5$

Sujeito a:

Cluster 1				Cluster 2						
$+x_1$				$+x_2$			$<=$	1		
	$+x_4$				$+x_3$		$<=$	1	Arestas entre *clusters*	
	$+x_4$					$+x_5$	$<=$	1		
			$+x_7$			$+x_5$	$<=$	1	↑	
$+x_1$	$+x_4$						$<=$	1	↓	
$+x_1$		$+x_6$					$<=$	1		
				$+x_2$	$+x_3$		$<=$	1		
					$+x_3$	$+x_5$	$<=$	1	Arestas intra *clusters*	
	$+x_4$	$+x_6$					$<=$	1		
	$+x_4$		$+x_7$				$<=$	1		
		$+x_6$	$+x_7$				$<=$	1		
$x_1,$	$x_4,$	$x_6,$	$x_7,$	$x_2,$	$x_3,$	x_5	$\in$	$\{0,1\}$		

O número de restrições entre *clusters* é igual a 4 e representam as primeiras linhas do conjunto de restrições de adjacências. Aplicando a relaxação lagrangeana sobre elas, obtém-se a seguinte função objetivo:

$$v\left(L_\lambda PMCIV\right) = Max \left\{ \begin{array}{l} x_1 + x_4 + x_6 + x_7 + x_2 + x_3 + x_5 + \\ \lambda_1 \left(1 - x_1 - x_2\right) + \lambda_2 \left(1 - x_3 - x_4\right) + \\ \lambda_3 \left(1 - x_4 - x_5\right) + \lambda_4 \left(1 - x_5 - x_7\right) \end{array} \right\} \tag{3.13}$$

que pode ser escrita como:

$$v\left(L_\lambda PMCIV\right) = Max \left\{ \begin{array}{l} \left(1 - \lambda_1\right) x_1 + \left(1 - \lambda_2 - \lambda_3\right) x_4 + x_6 + \left(1 - \lambda_4\right) x_7 + \\ \left(1 - \lambda_1\right) x_2 + \left(1 - \lambda_2\right) x_3 + \left(1 - \lambda_3 - \lambda_4\right) x_5 + \\ \left(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4\right) \end{array} \right\}$$

Assim, o problema relaxado pode ser formulado como:

$$v\left(L_\lambda PMCIV\right) = Max \left\{ \begin{array}{l} \left(1 - \lambda_1\right) x_1 + \left(1 - \lambda_2 - \lambda_3\right) x_4 + x_6 + \left(1 - \lambda_4\right) x_7 + \\ \left(1 - \lambda_1\right) x_2 + \left(1 - \lambda_2\right) x_3 + \left(1 - \lambda_3 - \lambda_4\right) x_5 + \\ \left(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4\right) \end{array} \right\} \tag{3.14}$$

Sujeito a:

$$\begin{aligned} & x_1 + x_4 \leq 1 \\ & x_1 + x_6 \leq 1 \\ & x_2 + x_3 \leq 1 \\ & x_3 + x_5 \leq 1 \\ & x_4 + x_6 \leq 1 \\ & x_4 + x_7 \leq 1 \\ & x_6 + x_7 \leq 1 \\ & x_1, x_4, x_6, x_7, x_2, x_3, x_5 \in \{0, 1\} \end{aligned} \tag{3.15}$$

Agora, o problema relaxado pode ser subdividido em dois subproblemas distintos: $v(L_\lambda PMCIV^1)$ e $v(L_\lambda PMCIV^2)$. Esses subproblemas estão definidos a seguir.

$$v\left(L_\lambda PMCIV^1\right) = Max\left\{(1-\lambda_1)\,x_1 + (1-\lambda_2-\lambda_3)\,x_4 + x_6 + (1-\lambda_4)\,x_7\right\} \tag{3.16}$$

Sujeito a:

$$\begin{aligned} & x_1 + x_4 \leq 1 \\ & x_1 + x_6 \leq 1 \\ & x_4 + x_6 \leq 1 \\ & x_4 + x_7 \leq 1 \\ & x_6 + x_7 \leq 1 \\ & x_1, x_4, x_6, x_7 \in \{0, 1\} \end{aligned} \tag{3.17}$$

$$v\left(L_\lambda PMCIV^2\right) = Max\left\{(1-\lambda_1)\,x_2 + (1-\lambda_2)\,x_3 + (1-\lambda_3-\lambda_4)\,x_5\right\} \tag{3.18}$$

Sujeito a:

$$\begin{aligned} & x_2 + x_3 \leq 1 \\ & x_3 + x_5 \leq 1 \\ & x_2, x_3, x_5 \in \{0, 1\} \end{aligned} \tag{3.19}$$

Com isso o problema relaxado 3.14-3.15 pode ser escrito como:

$$v\left(L_\lambda PMCIV\right) = \sum_{p=1}^{2} v\left(L_\lambda PMCIV^p\right) + (\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4) \tag{3.20}$$

Pela Equação 3.20 observa-se que os subproblemas gerados podem ser resolvidos seqüencialmente ou de forma paralelizada. Ao se paralelizar, o tempo computacional pode ser reduzido.

3.4 Considerações finais

Este capítulo procurou mostrar o PMCIV e a relaxação lagrangeana com clusters, assim como um exemplo de sua aplicação. Percebe-se que a LagClus pode ser mais ou menos "forte" conforme o particionamento. Quanto menor for $\overline{P}$, mais forte será a LagClus pois menos restrições serão relaxadas, enquanto que, quanto maior for $\overline{P}$, mais próximo a LagClus está da relaxação lagrangeana sobre todas as arestas de G, pois os *clusters* tendem a ser menores. No limite, quando $\overline{P} = |V|$, cada vértice torna-se um *cluster* e aí todas as arestas de G são relaxadas.

Com isso, é importante variar o valor de $\overline{P}$ de modo a obter uma relaxação satisfatória para o problema em questão, não existindo ainda uma formulação que indique o melhor valor para esse parâmetro em função de $|V|$ e $|E|$.

A LagClus foi aplicada com sucesso ao problema da rotulação cartográfica. Ela também foi aplicada ao problema de carregamento de paletes e provou ser mais forte do que as relaxações convencionais para esse problema. Esses dois resultados serão mostrados nos próximos capítulos.

CAPÍTULO 4

ROTULAÇÃO CARTOGRÁFICA DE PONTOS

O Problema da Rotulação Cartográfica de Pontos (PRCP) consiste em rotular os pontos de um mapa evitando as sobreposições (conflitos) dos rótulos. Sendo assim, esse problema possui forte ligação com o PMCIV, pois consiste em obter o máximo conjunto possível de pontos rotulados, sem sobreposição. Sob esse enfoque, esse problema pode ser modelado exatamente como um PMCIV e com isso, alguns pontos podem ficar sem seus respectivos rótulos. Entretanto, sob o ponto de vista cartográfico, é interessante que todos os pontos sejam rotulados, mesmo que algumas sobreposições sejam inevitáveis. Com isso, existem na literatura duas outras abordagens para esse problema: uma que considera o Problema do Máximo Número de Rótulos Sem Conflitos (PMNRSC) e outra que considera o Problema da Minimização do Número de Conflitos (PMNC). O PMNRSC ainda não apresenta uma modelagem matemática de programação inteira binária, sendo os melhores resultados encontrados por Yamamoto e Lorena (2005) usando Algoritmo Genético Construtivo (Lorena e Furtado, 2001). Já o PMNC possui duas modelagens matemáticas: uma baseada nas arestas do grafo de conflito e outra baseada nos pontos a serem rotulados. Essas duas formulações foram propostas por Ribeiro e Lorena (2004c,a).

O PMNRSC procura maximizar o número de pontos rotulados sem conflitos não se importando muito com o aspecto visual do mapa, podendo apresentar áreas ilegíveis. No entanto, o PMNC procura rotular todos os pontos, minimizando os conflitos gerados. Essa última abordagem é interessante, pois, se em um mapa os conflitos são inevitáveis, esses poderão ser "espalhados" pelo mapa, evitando assim áreas ilegíveis.

4.1 Padronização cartográfica e grafo de conflito

A Figura 4.1 apresenta parte de um mapa das estações ferroviárias brasileiras. Pode-se observar nas áreas indicadas pelas setas, conflitos entre os rótulos dos pontos, o que torna o mapa ilegível. No entanto, os rótulos devem estar próximos do ponto, obedecendo a uma padronização cartográfica o que limita as possíveis posições dele.

A padronização cartográfica por meio eletrônico mais conhecida é a de Christensen et al. (1995). Ao se definir essas possíveis localizações do rótulo, este problema pode ser formulado como um problema de otimização combinatória. A Figura 4.2 mostra um conjunto de 8 possíveis posições para o rótulo de um ponto, que também são conhecidas por posições candidatas. O número presente em cada posição, determina a preferência cartográfica sendo a posição um, a de maior interesse para a cartografia.

Considere um problema em que para cada ponto existem somente 4 posições candidatas, como mostra a Figura 4.2 (a). O mesmo pode ser facilmente representado por um grafo de conflitos. Sendo NP o número de pontos a serem rotulados e PC o número de posições candidatas de cada ponto, seja $G = \{V, E\}$ o grafo de conflito tal que $V = \{v_1, v_2, ..., v_{NP*PC}\}$ representa o conjunto das posições candidatas (vértices) e $E = \{(v_i, v_j) : i, j \in V, i \neq j\}$ as sobreposições (conflitos). A Figura 4.3(b) apresenta o grafo de conflito obtido para o problema da Figura 4.3(a), e na Figura 4.3(c) a solução ótima para esse problema. Para se determinar a qualidade de uma solução, é usado na literatura o percentual de rótulos livres (sem conflitos). No caso da Figura 4.3(c) têm-se 100% de rótulos livres.

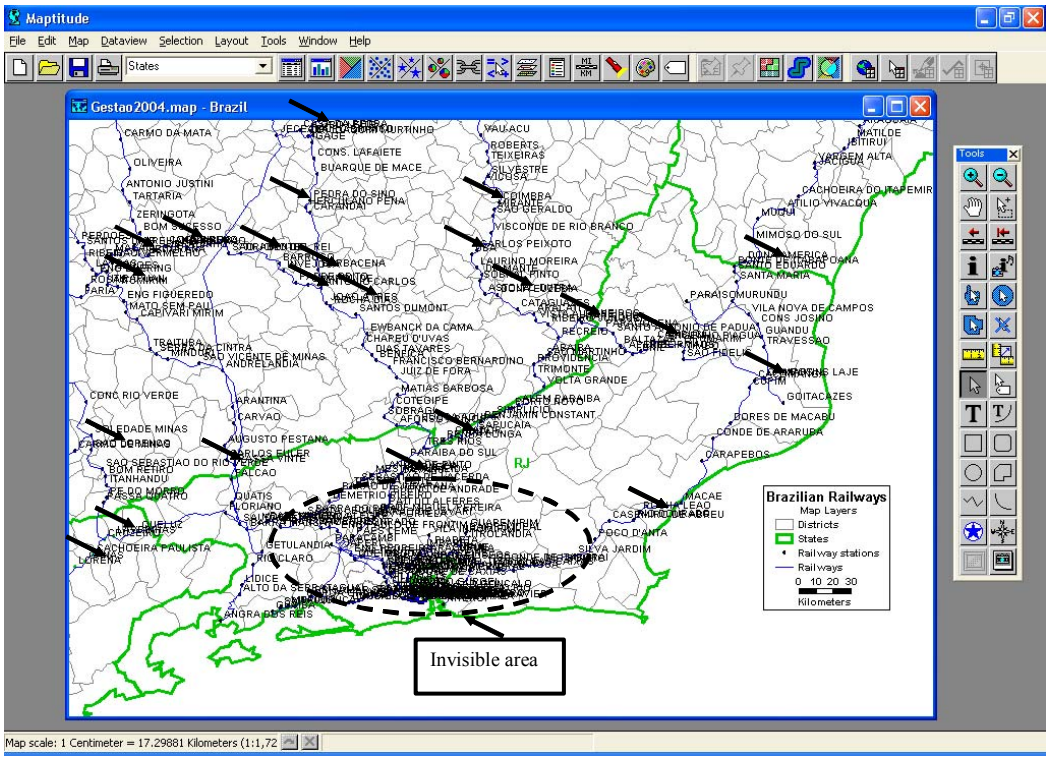


FIGURA 4.1 – Mapa das Estações Ferroviárias Brasileiras. As setas indicam conflitos de rótulos.

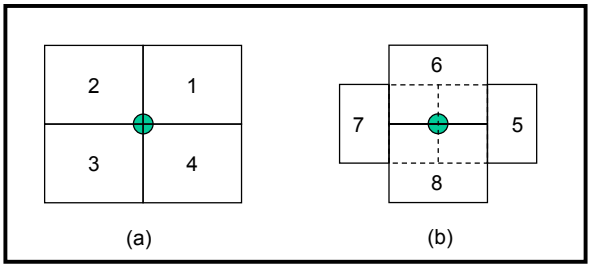


FIGURA 4.2 – Conjunto de 8 posições candidatas para rotular um ponto (Christensen et al., 1995).

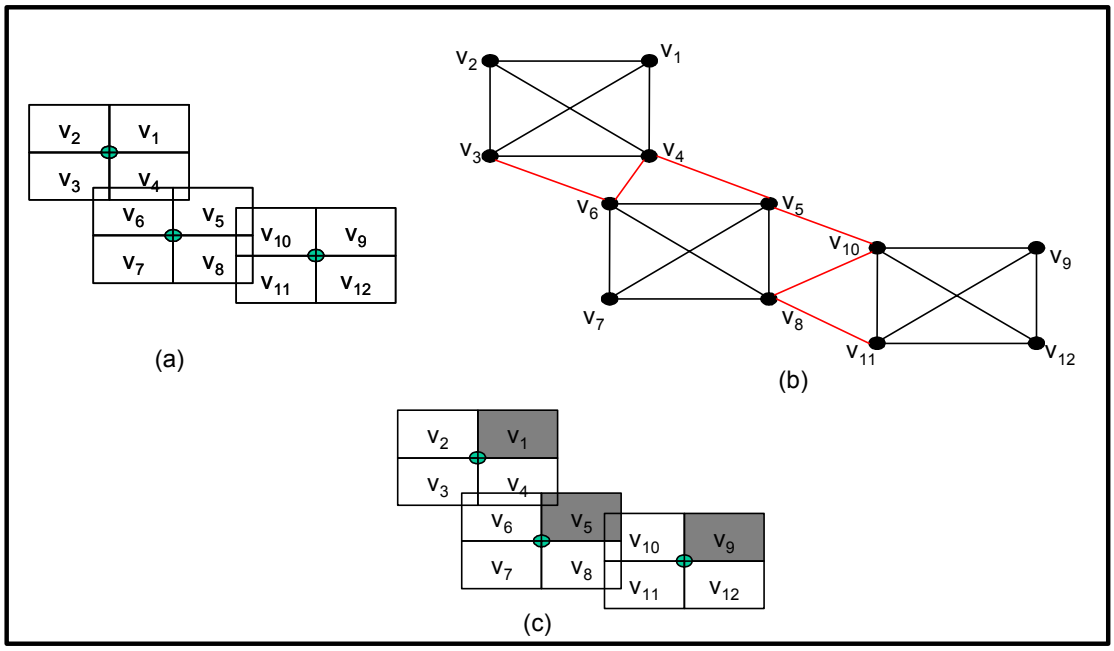


FIGURA 4.3 – Representação em grafo do PRCP. (a) Problema, (b) grafo relacionado e (c) solução ótima.

4.2 Revisão bibliográfica para o PRCP

A primeira abordagem consiste em tratar o PRCP como um PMCIV, desta forma, vários trabalhos propuseram diferentes algoritmos, assim como técnicas que permitem reduzir o número de restrições. Zoraster (1986, 1990, 1991) formulou matematicamente o PRCP porém, para lidar com as restrições de conflito, criou posições candidatas fictícias de custo elevado, de tal modo que se nenhuma das reais posições candidatas pudessem ser usadas para posicionar um rótulo, a posição fictícia era então utilizada. Em seu trabalho o autor apresentou uma relaxação lagrangeana obtendo alguns resultados para instâncias pequenas. Por outro lado, Strijk et al. (2000) propuseram uma modelagem de programação inteira binária que usa restrições de corte para reduzir o número de restrições de conflito presentes no modelo. Essas técnicas de redução apareceram antes nos trabalhos de Moon e Chaudhry (1984) e Murray e Church (1997a), e consistem em utilizar conceitos de cliques para transformar algumas restrições em uma única, reduzindo assim, o número total de restrições do problema original. Strijk et al. (2000) utilizaram uma relaxação de programação linear e aplicaram um algoritmo *Branch and Bound* para encontrar as soluções ótimas para os problemas testados. Devido às dificuldades encontradas, os autores ainda aplicaram e propuseram várias heurísticas, sendo elas: *Simulated Annealing*, Busca em Vizinhança Diversificada, *k-Opt* e *Busca Tabu*. Sendo a Busca Tabu o algoritmo que apresentou os melhores resultados.

Considerando a segunda abordagem ou seja, o PRCP como um PMNRSC, Christensen

et al. (1993, 1995) propuseram um método denominado Busca Exaustiva que faz uma procura por soluções melhores alternando as posições de rótulos previamente posicionados. Christensen et al. (1995) também propuseram um Algoritmo Guloso com sucessivas otimizações locais e um algoritmo denominado *Discrete Gradient Descent* que considera as posições alternativas dos rótulos, porém, esse algoritmo, apesar de rápido, tem dificuldades para escapar de mínimos locais.

Verner et al. (1997) aplicou um algoritmo genético com máscara no PRCP. Eles propuseram uma maneira de trabalhar com as máscaras de modo que, se um rótulo está em conflito, será permitido a troca de posições através de cruzamentos e mutações. Mais tarde, Yamamoto et al. (2002) propuseram um algoritmo de Busca Tabu eficiente que forneceu resultados muito bons quando comparados com a literatura.

Schreyer e Raidl (2004) utilizaram um Sistema de Colônia de Formigas (*Ant Colony System-ACS*) mas seus resultados não foram satisfatórios quando comparados com os obtidos por Yamamoto et al. (2002). Yamamoto e Lorena (2005) desenvolveram um algoritmo exato para o PMNRSC e aplicaram o Algoritmo Genético Construtivo (AGC) proposto por Lorena e Furtado (2001). O algoritmo exato foi aplicado a instâncias de até 25 pontos, pois utiliza uma estrutura em árvore o que limitou a sua aplicação. Em contrapartida, o AGC foi aplicado a instâncias com até 1000 pontos, fornecendo bons resultados até então.

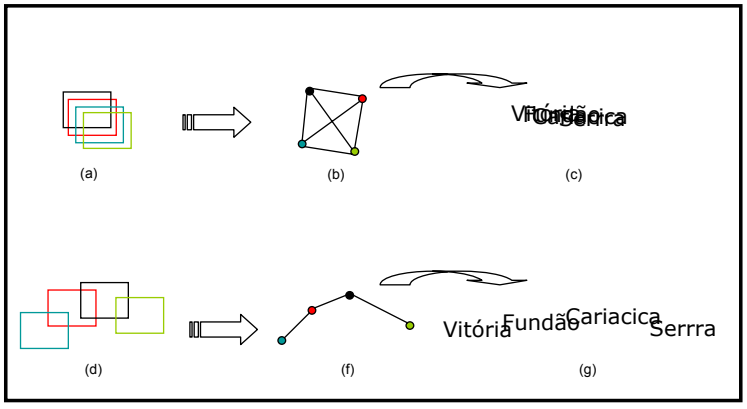


FIGURA 4.4 – Comparação entre abordagens.

A terceira abordagem que considera o PRCP como um PMNC, foi introduzida recentemente por Ribeiro e Lorena (2004d,e). Ela considera a minimização do número de conflitos e, em alguns casos, as soluções encontradas são limitantes inferiores para a segunda abordagem (PRCP como um PMNRSC). Considerando a Figura 4.4, as soluções mostradas em (a) e (d), sob o ponto de vista da abordagem dado pelo PMNRSC,

são iguais pois apresentam o mesmo número de rótulos em conflitos (4), porém se considerarmos a representação em grafo mostrada na figura (ver Figura 4.4(b) e (f)), a solução (d) é mais interessante que a mostrada em (a), pois apresenta um número de arestas (conflitos) menor que a solução (a). Isso corresponde à idéia por trás do PMNC. Porém, o PMNRSC pode ser um limitante superior para o PMNC (Ribeiro e Lorena, 2004b) quando comparadas suas soluções ótimas para um mesmo problema. No entanto, o PMNC, no caso de conflitos inevitáveis, permite "espalhar" mais os rótulos, fornecendo uma melhor visualização do mapa como pode ser visto na Figura 4.4(g).

Como dito anteriormente, existem duas modelagens para o PMNC que diferem, basicamente, no número de restrições de conflitos (adjacências) geradas. A que trabalha com pontos foi inspirada no trabalho de Murray e Church (1997a), e reduz consideravelmente, o número de restrições quando comparadas com a formulação baseada em arestas. A LagClus aplicada sobre essa formulação, mesmo com objetivo diferente do usado no trabalho de Yamamoto e Lorena (2005), quando traduzida para percentual de rótulos livres, apresentou resultados melhores que os obtidos por eles.

4.3 Modelagem matemática baseada em arestas

A modelagem matemática baseada em arestas tem como objetivo, como dito anteriormente, minimizar o número de conflitos. Para cada ponto i existe um número fixo PC_i de posições candidatas. Cada posição candidata é representada por uma variável binária $x_{i,j}$ onde $i = 1..NP$, sendo NP o número de pontos a serem rotulados; e $j = 1, ..., PC_i$. Se $x_{i,j} = 1$ a posição candidata j do ponto i será alocada (receberá o rótulo do ponto i), caso contrário, $x_{i,j} = 0$. Além disso, a cada posição candidata é associado um custo (uma preferência cartográfica) representado por $w_{i,j}$.

Cada posição candidata $x_{i,j}$ possui um conjunto $S_{i,j}$ de pares de índices de posições candidatas conflitantes com $x_{i,j}$. $S_{i,j}$ é um conjunto de pares de índices (k,t) de posições candidatas $x_{k,t}$ que são conflitantes com $x_{i,j}$. Para todo $(k,t) \in S_{i,j}$, onde $k \in \{1, ..., NP\} : k > i$ e $t \in \{1, ..., PC_k\}$, existe uma variável binária $y_{i,j,k,t}$ que representa o conflito, uma aresta no grafo G.

Assim, considerando as informações acima, o Problema de Minimização do Número de Conflitos baseado em arestas ($PMNC^a$), pode ser representado por:

$$v\left(PMNC^a\right) = Min \sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \left(w_{i,j} x_{i,j} + \sum_{(k,t) \in S_{i,j}} y_{i,j,k,t} \right) \tag{4.1}$$

Sujeito a:

$$\sum_{j=1}^{PC_i} x_{i,j} = 1 \qquad \forall i = 1...NP \tag{4.2}$$

$$\begin{aligned} x_{i,j} + x_{k,t} - y_{i,j,k,t} \leq 1 \qquad & \forall i = 1...NP \\ & \forall j = 1...PC_i \\ & (k,t) \in S_{i,j} \end{aligned} \tag{4.3}$$

$$\begin{aligned} x_{i,j}, x_{k,t} \quad \text{e} \quad y_{i,j,k,t} \in \{0,1\} \qquad & \forall i = 1...NP \\ & \forall j = 1...PC_i \\ & (k,t) \in S_{i,j} \end{aligned} \tag{4.4}$$

A restrição 4.2 garante que para cada ponto i uma, e somente uma, posição candidata receberá o rótulo de i. A restrição 4.3 garante que se duas posições candidatas $x_{i,j}$ e $x_{k,t}$ conflitantes forem alocadas, a função objetivo será penalizada com o surgimento da variável $y_{i,j,k,t}$, para que a restrição $x_{i,j} + x_{j,k} - y_{i,j,k,t} \leq 1$ seja válida.

Então, deve-se observar que as variáveis de conflitos $y_{i,j,k,t}$ aparecem somando na função objetivo, que, ao ser minimizada, deverá procurar eliminá-las ou minimizá-las (caso a eliminação não seja possível).

4.3.1 Relaxações consideradas para a modelagem baseada em arestas

O modelo definido em 4.1-4.4 é parecido com o proposto por Zoraster (1990), porém ele permite posicionar todos os rótulos minimizando o número de conflitos.

Entretanto, essa formulação quando aplicada a problemas com até 500 pontos forneceu, com auxílio do CPLEX 7.5 (ILOG, 2001), as soluções ótimas em poucos segundos. Em contrapartida, para instâncias maiores com 750 e 1000 pontos resolvidas por Yamamoto et al. (2002) e Yamamoto e Lorena (2005), os resultados ótimos não foram obtidos. Assim, algumas relaxações foram propostas de modo a permitir resolver o $PMNC^a$.

Semelhante ao que foi feito por Zoraster (1990), a primeira relaxação considerada foi uma relaxação lagrangeana sobre o conjunto de restrições definidos em 4.3. Assim, para um dado vetor de multiplicadores $\lambda \in R_+^C$, onde C é o número de restrições de conflito definido como:

$$C = \sum_{i=1}^{NP} \sum_{j=1}^{PC_i} |S_{i,j}| \tag{4.5}$$

uma relaxação lagrangeana para o $PMNC^a$ (por questões de notação, considere P como sendo $PMNC^a$) pode ser definida como sendo:

$$v(L_\lambda P) = Min \sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \left(w_{i,j} x_{i,j} + \sum_{(k,t) \in S_{i,j}} y_{i,j,k,t} \right) + \sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \sum_{(k,t) \in S_{i,j}} \lambda_{i,j,k,t} \left(x_{i,j} + x_{k,t} - y_{i,j,k,t} \right) - \sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \sum_{(k,t) \in S_{i,j}} \lambda_{i,j,k,t} \tag{4.6}$$

Sujeito a 4.2 e 4.4.

O valor ótimo $v\left(L_\lambda P\right)$ é menor ou igual a $v(P)$ e resulta da solução do problema *dual* lagrangeano $\underset{\lambda \geq 0}{Max} \left\{ v\left(L_\lambda P\right) \right\}$, como foi visto no Capítulo 3. Com isso, foi considerado o algoritmo de subgradientes descrito por Narciso e Lorena (1999). Em uma iteração l, os subgradientes são definidos como $g^{\lambda_l} = x_{i,j}^{\lambda_l} + x_{k,t}^{\lambda_l} - y_{i,j,k,t}^{\lambda_l} - 1$, $i \in \{1, ..., NP\}$, $j \in \{1, ..., PC_i\}$ e $(k,t) \in S_{i,j}$, tal que $\left(x^{\lambda_l}, y^{\lambda_l}\right)$ está na solução ótima de $\left(L^\lambda P\right)$. O método de subgradientes atualiza o multiplicador λ_l fazendo $\lambda_{l+1} = \lambda_l + \theta_l g^{\lambda_l}$ onde θ_l é o tamanho da passo calculado como $\theta_l = \frac{\pi(ub_l - lb_l)}{\left\| g^{\lambda_l} \right\|^2}$, sendo ub_l a melhor solução factível encontrada até a iteração l, e lb_l o melhor limitante dual encontrado também até a iteração l. O controle utilizado para o parâmetro π é o mesmo proposto por Held e Karp (1971). Inicia-se com o valor 2 e se durante 15 iterações o valor do limite superior (ub_l) não decrescer, π é dividido pela metade. As condições de parada consideradas são:

1)$\pi \leq 0.005$

2)$(ub_l - lb_l) < 1$

3) $\left\| g^\lambda \right\|^2 = 0$

Uma solução relaxada $Sol_{rel} = \left(x^\lambda, y^\lambda\right)$ de $(L^\lambda P)$, composta das variáveis de posições candidatas $\left(x_{i,j}^\lambda\right)$ e de variáveis de conflito $\left(y_{i,j,k,t}^\lambda\right)$, também é uma solução factível para P pois o conjunto de restrições 4.2 é respeitado. Essa solução factível é melhorada utilizando a heurística de busca local mostrada na Figura 4.5.

A segunda relaxação testada foi a Lagrangeana*Surrogate* (*LagSur*) (Narciso e Lorena, 1999). Novamente o conjunto de restrições relaxadas foi o definido em 4.3. Primeiramente é feita a relaxação no sentido *Surrogate* como descrito por Glover (1968) sobre 4.3, e em seguida no sentido lagrangeano. Assim, para um dado vetor de multiplicadores $\lambda \in R_+^C$, uma relaxação *Surrogate* para P pode ser encontrada. Em seguida, ao se relaxar a restrição *surrogate* encontrada utilizando um multiplicador dual $t \geq 0$, pode-se obter

a seguinte relaxação *LagSur* para o problema P:

$$v(L_tSP_1^{\lambda}) = Min \sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \left(w_{i,j}x_{i,j} + \sum_{(k,t)\in S_{i,j}} y_{i,j,k,t} \right) \\ +t \left(\sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \sum_{(k,t)\in S_{i,j}} \lambda_{i,j,k,t} \left(x_{i,j} + x_{k,t} - y_{i,j,k,t} \right) - \sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \sum_{(k,t)\in S_{i,j}} \lambda_{i,j,k,t} \right) \tag{4.7}$$

Sujeito a 4.2 e 4.4.

Heurística de Melhoria - HM
1. Para cada elemento da Solução factível, armazene em um vetor de conflito o número de conflitos para cada posição.
2. Para *i=1* até o tamanho do vetor de conflitos;
  3. Se Vetor de conflitos[*i*] ≠ 0
    4. Procurar dentre as possíveis posições candidatas *j*, a que apresenta o menor número de conflitos com a solução viável atual.
    5. Se houver alguma posição candidata *j* com número de conflitos inferior a Vetor de conflitos[*i*], trocar Solução factível[*i*] por posição candidata *j*.
6. Fim do Para.

FIGURA 4.5 – Heurística de Melhoria (HM) aplicada no algoritmo de subgradientes.

Uma característica interessante dessa relaxação é que para $t = 1$ obtém-se a relaxação lagrangeana mostrada anteriormente. Para o vetor λ fixo, o melhor valor para t pode ser calculado resolvendo-se o seguinte problema dual:

$$v\left(D_t^{\lambda}\right)_1 = \underset{t\geq 0}{Max}\, v\left(L_tSP_1^{\lambda}\right) \tag{4.8}$$

O valor ótimo $v\left(D_t^{\lambda}\right)_1$ é um limitante melhor que o obtido pela relaxação lagrangeana (Senne e Lorena, 2000). Senne e Lorena (2000) descrevem um algoritmo de busca dicotômica que aproxima o melhor valor de t. Sendo t^* esse melhor valor, se para um dado número de iterações do algoritmo de subgradiente o valor de t^* repete, então esse valor é mantido fixo até o final do procedimento e a busca não é mais executada. Para a iteração l, os multiplicadores são atualizados como $\lambda_{l+1} = \lambda_l + \theta_l g^{\lambda_l}$ onde θ_l é o tamanho do passo calculado por $\theta_l = \frac{\pi(ub_l - lb_l)}{\left\| g^{\lambda_l} \right\|^2}$, sendo ub_l e lb_l, respectivamente, a melhor solução factível dada pela heurística HM, e o melhor limitante dual, ambos obtidos até a iteração l.

A relaxação *LagSur* também foi aplicada sobre o conjunto de restrições 4.2. Da

mesma forma, dado um vetor de multiplicadores $\lambda \in R^{NP}$, obtém-se uma relaxação *Surrogate* para P que pode ser relaxada novamente no sentido lagrangeano utilizando um multiplicador dual t irrestrito, obtendo assim uma nova relaxação *LagSur* que pode ser expressa como:

$$v(L_tSP_2^{\lambda}) = Min \sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \left(w_{i,j}x_{i,j} + \sum_{(k,t)\in S_{i,j}} y_{i,j,k,t} \right) + t \left(\sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \lambda_i x_{i,j} - \sum_{i=1}^{NP} \lambda_i \right) \quad (4.9)$$

Sujeito a 4.3 e 4.4.

Agora, os subgradientes são $g^{\lambda_l} = \sum_{j=1}^{PC_i} x_{i,j}^{\lambda_l} - 1 \; \forall i = 1, ..., NP$, e o método de subgradientes atualiza os multiplicadores λ_l como mostrado anteriormente, $\lambda_{l+1} = \lambda_l + \theta_l g^{\lambda_l}$, onde θ_l é o tamanho do passo calculado por $\theta_l = \frac{\pi(ub_l - lb_l)}{\left\| g^{\lambda_l} \right\|^2}$, sendo ub_l a melhor solução factível, até a iteração l, encontrada pela heurística HM executada após uma heurística de construção (HC) (ver Figura 4.6); e lb_l o melhor limitante inferior encontrado até a iteração l, dado pelo dual lagrangeano.

Heurística de Construção - HC
1. Preencher Vetor solução factível com zeros;
2. Para *i=1 até NP*
   3. Procurar em solução relaxada todas as posições diferentes de zero para o ponto *i*;
   4. Selecionar para solução factível na posição *i*, a posição candidata *j* com o menor número de conflitos com os elementos da solução factível corrente. Em caso de igualdade, selecionar aquela que possui o menor conjunto $S_{i,j}$.
   5. Se nenhuma posição candidata *j* para o ponto *i* está na solução relaxada, escolher a posição candidata que apresentar o menor conjunto $S_{i,j}$.
6. Fim do Para.

FIGURA 4.6 – Heurística de Construção (HC) aplicada no algoritmo de subgradientes.

A heurística de construção constrói uma solução factível a partir da solução relaxada $Sol_{rel} = \left(x^{\lambda}, y^{\lambda}\right)$ de $\left(L_{t^*}SP_2^{\lambda}\right)$, encontrando posições candidatas (restrição 4.2) com o menor número de conflitos com a solução factível que está sendo construída. Se algum ponto i não está rotulado, a heurística procura dentre todas as possíveis posições candidatas j, aquela que apresenta o menor conjunto de conflitos definidos em $S_{i,j}$. A solução factível obtida é então melhorada usando a heurística HM.

Avaliando agora a idéia desta proposta de tese, considere a Figura 4.7. O grafo obtido

para o problema mostrado é particionado em duas partes, formando dois *clusters*. Ao se dividir, dois subproblemas são formados, menores do que o problema original.

A LagClus pode ser então aplicada da seguinte maneira:

1) Aplicar uma heurística de particionamento para dividir G em $\overline{P}$ partes, formando $\overline{P}$ *clusters*;

2) Dois multiplicadores distintos relaxam no sentido lagrangeano, as restrições definidas em 4.2 e o grupo das restrições de conflito que correspondem às arestas entre *clusters*;

3) A relaxação lagrangeana é decomposta em $\overline{P}$ subproblemas e resolvida.

Observa-se que o passo 2 não é exatamente aquele descrito na Seção 3.2 pois nesse caso, as restrições definidas em 4.2 também devem ser relaxadas, pois, como pode ser visto na Figura 4.7, existem posições candidatas de um mesmo ponto em *clusters* diferentes.

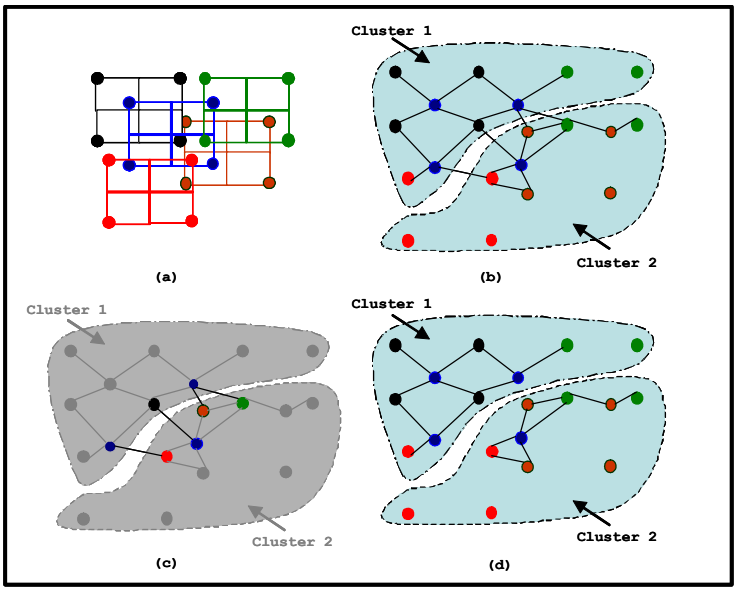


FIGURA 4.7 – Particionamento do grafo de conflito. (a) Problema, (b) grafo de conflitos e os *clusters*, (c) arestas entre *clusters*, e (d) subproblemas gerados.

4.3.2 Resultados computacionais para a modelagem baseada em arestas

Os testes computacionais foram feitos sobre as instâncias propostas por Yamamoto e Lorena (2005) disponíveis em http://www.lac.inpe.br/~lorena/instancias.html. O programa, em C++, e os testes, foram feitos em um computador com um processador Pentium IV 2.66 GHz e 512 MB de memória RAM. Como realizado em Zoraster (1990); Yamamoto et al. (2002); Yamamoto e Lorena (2005), para todas os problemas não foram

consideradas preferências cartográficas, ou seja, considera-se custo ou penalidade igual a 1 para todas as posições candidatas, sendo o número dessas posições igual a 4.

As Tabelas 4.1, 4.2 e 4.3 apresentam os resultados médios obtidos para $(L_\lambda P)$ e $(L_\lambda SP_1^\lambda)$ (problemas com 25 pontos possuem 8 instâncias, os demais, 25 instâncias). As colunas indicam:

- Problema - Número de pontos que devem ser rotulados;
- Solução Ótima - Solução ótima do problema definido em 4.1-4.4, obtida usando o CPLEX 7.5;
- Limite Inferior - Melhor limite inferior (dual) encontrado;
- Limite Superior - Melhor limite superior (solução factível) encontrado;
- GAP UB - Desvio percentual do melhor limite superior encontrado em relação à solução ótima, calculado como: $Gap\_ub = \left(\frac{LimiteSuperior - SolucaoOtima}{SolucaoOtima}\right) * 100$;
- GAP LB - Desvio percentual do melhor limite superior encontrado em relação à solução ótima, calculado como: $Gap\_ub = \left(\frac{SolucaoOtima - LimiteInferior}{SolucaoOtima}\right) * 100$;
- Iterações - Número de iterações utilizadas pelo algoritmo de subgradientes;
- Tempo - Tempo computacional total em segundos.

TABELA 4.1 – Resultados médios para o limitante inferior das relaxações lagrangeana e LagSur sobre as restrições 4.3.

Problema	**Solução ótima**	**Limite inferior**				**GAP LB (%)**			
		$(L^\lambda P)$	$(L_\lambda SP_1^\lambda)$ $t = 0,25$	$t = 0,50$	$t = 0,75$	$(L^\lambda P)$	$(L_\lambda SP_1^\lambda)$ $t = 0,25$	$t = 0,50$	$t = 0,75$
25	27,75	25,00	24,96	25,00	25,00	9,72	9,87	9,72	9,72
100	100,00	NC	NC	NC	NC	NC	NC	NC	NC
250	250,00	NC	NC	NC	NC	NC	NC	NC	NC
500	500,84	498,20	493,46	496,91	499,45	0,53	1,48	0,80	0,28
750	-	749,98	743,42	748,74	749,98	-	-	-	-
1000	-	999,93	997,04	999,93	999,95	-	-	-	-

A procura pelo melhor multiplicador t descrita por Senne e Lorena (2000) conduziu ao valor zero em $(L_\lambda SP_1^\lambda)$. Esse fato já era esperado pois toda informação sobre os conflitos (variáveis $y_{i,j,k,t}$) foi relaxada. Com isso, alguns valores para t foram pré-fixados: 0,25; 0,5 e 0,75.

Para os problemas com 750 e 1000 pontos, como dito anteriormente, as soluções ótimas não foram obtidas. Instâncias com 100 e 250 pontos são simples, uma solução totalmente

sem conflito é encontrada rapidamente, com isso o algoritmo de subgradientes pára em limitantes inferiores ruins, conseqüentemente os limitantes inferiores e os *Gaps* não foram calculados, sendo substituídos por *NC* (Não calculado). A mesma situação ocorreu em algumas instâncias com 500 pontos. Com isso, os resultados mostrados nas Tabelas 4.1, 4.2 e 4.3 correspondem aos problemas em que o processo de otimização encontra uma das condições de parada do algoritmo de subgradientes.

Considerando a Tabela 4.1, os *Gaps* do limitante inferior para $(L_\lambda P)$ variaram de 0,53% a 9,72%; e para $(L_\lambda SP_1^\lambda)$, no melhor caso, de 0,28% a 9,72%. Na Tabela 4.2, os *Gaps* encontrados para o limitante superior variaram de 0,00% a 4,14% e de 0,00% a 3,21% para $(L_\lambda P)$ e $(L_\lambda SP_1^\lambda)$, respectivamente. $(L_\lambda SP_1^\lambda)$ apresentou resultados melhores para os dois *Gaps* avaliados. Os tempos computacionais variaram de 0 a 19,60s para ambas as relaxações.

TABELA 4.2 – Resultados médios para o limitante superior das relaxações lagrangeana e LagSur sobre as restrições 4.3.

Problema	**Solução ótima**	**Limite superior**				**GAP UB (%)**			
		$(L^\lambda P)$	$(L_\lambda SP_1^\lambda)$ $t=0,25$	$t=0,50$	$t=0,75$	$(L^\lambda P)$	$(L_\lambda SP_1^\lambda)$ $t=0,25$	$t=0,50$	$t=0,75$
25	27,75	28,88	28,38	28,25	28,63	4,14	2,33	1,82	3,21
100	100,00	100,00	100,00	100,00	100,00	0,00	0,00	0,00	0,00
250	250,00	250,00	250,00	250,04	250,00	0,00	0,00	0,02	0,00
500	500,84	503,92	502,88	503,16	503,80	0,61	0,41	0,46	0,59
750	-	774,44	771,80	772,68	774,44	-	-	-	-
1000	-	1086,44	1075,64	1079,24	1086,40	-	-	-	-

TABELA 4.3 – Iterações e tempos computacionais das relaxações lagrangeana e LagSur sobre as restrições 4.3.

Problema	**Número de iterações**				**Tempo (s)**				
	$(L^\lambda P)$	$(L_\lambda SP_1^\lambda)$ $t=0,25$	$t=0,50$	$t=0,75$	**Solução ótima**	$(L^\lambda P)$	$(L_\lambda SP_1^\lambda)$ $t=0,25$	$t=0,50$	$t=0,75$
25	146,63	148,88	148,38	146,63	1,60	0,25	0,13	0,13	0,13
100	1,16	1,16	1,16	1,16	0,02	0,00	0,00	0,00	0,04
250	2,48	2,36	7,72	2,68	0,06	0,00	0,00	0,04	0,00
500	146,00	140,24	134,48	146,20	3,12	2,76	2,64	2,64	2,88
750	146,00	146,00	146,00	146,00	-	8,48	8,44	8,40	8,56
1000	146,00	146,00	146,00	146,00	-	19,32	19,60	19,52	19,08

Os resultados obtidos para $(L_\lambda SP_2^\lambda)$ estão mostrados na Tabela 4.4. O CPLEX 7.5 foi utilizado para resolver os problemas de programação linear inteira binária. Essa relaxação é mais "forte" que $(L_\lambda P)$ e $(L_\lambda SP_1^\lambda)$ no entanto, os resultantes não foram interessantes. Somente o limite inferior foi melhorado, e os tempos computacionais aumentaram

consideravelmente. Os limitantes superiores não foram melhores, e os problemas com 500, 750 e 1000 pontos não puderam ser resolvidos em 7200 segundos de execução do algoritmo de subgradientes.

A última relaxação aplicada foi a LagClus. A Tabela 4.5 mostra as informações referentes ao particionamento para cada classe de problemas. A primeira coluna diz respeito ao problema, seguida pelo número de posições candidatas, número de vértices presentes no grafo de conflito, número de *clusters* considerado, e por último de forma aproximada, o número de vértices presentes em cada *cluster*.

TABELA 4.4 – Iterações e tempos computacionais das relaxações lagrangeana e LagSur sobre as restrições 4.2.

Problema	**Solução ótima**	**Limite inferior**	**Limite superior**	**GAP LB %**	**GAP UB %**	**Iterações**	**Tempo (s)**	
							Solução ótima	$(L_\lambda SP_2^\lambda)$
25	27,75	25,13	28,38	9,27	2,29	151,63	1,60	104,63
100	100,00	NC	100,00	NC	0,00	1,00	0,02	0,16
250	250,00	NC	250,00	NC	0,00	1,48	0,06	0,92

TABELA 4.5 – Informações sobre os particionamentos realizados.

Problema	**Número de posições candidatas**	**Número de vértices**	**Número de *clusters***	**Número médio de vértices em cada *cluster***
25	4	100	2	50
100	4	400	4	100
250	4	1000	10	100
500	4	2000	20	100
750	4	3000	25	120
1000	4	4000	60	66,67

TABELA 4.6 – Resultados médios obtidos com a LagClus.

Problema	**Solução ótima**	**Limite inferior**	**Limite superior**	**GAP LB %**	**GAP UB %**	**Iterações**	**Tempo(s)**	
							Solução ótima	**LagClus**
25	27,75	25,13	27,88	9,27	0,46	148,50	1,60	23,88
100	100,00	NC	100,00	NC	0,00	1,00	0,02	0,16
250	250,00	NC	250,00	NC	0,00	7,12	0,06	2,36
500	500,84	497,30	501,52	0,76	0,14	103,16	3,12	82,72
750	-	749,41	767,08	-	-	145,28	-	337,80
1000	-	1002,11	1070,60	-	-	145,96	-	817,00

Na Tabela 4.6 estão mostrados os resultados obtidos com a LagClus. O particionamento foi feito usando o METIS de Karypis e Kumar (1998c). Os maiores problemas com 750 e 1000 pontos, foram resolvidos em média em 337,80 e 817,00s, respectivamente. Pode-se

observar que os limitantes inferiores e superiores foram melhores que os obtidos pelas demais relaxações, apenas o tempo foi inferior. Os *Gaps* superiores variaram de 0,00% a 0,46%, e os inferiores de 0,76% a 9,27%; e em algumas instâncias, o limitante trivial imposto pelo número de pontos, foi ultrapassado. Novamente os problemas com 100 e 250 pontos foram resolvidos rapidamente (uma solução sem conflito é obtida) e com isso, o algoritmo de subgradientes pára em limitantes inferiores ruins. Nesses casos, os *Gaps* não são calculados (NC).

Como foi previsto pelo Capítulo 3, a LagClus é uma relaxação "forte", variável em função do número de *clusters*, que permite obter soluções melhores porém em um tempo computacional maior.

4.4 Modelagem matemática baseada em pontos

Considerando as definições da Seção 4.3 e, dado que para todo ponto i apenas umas das posições candidatas receberá o rótulo (restrições definidas em 4.2), as restrições de conflitos podem ser representadas entre pontos ao invés de entre posições candidatas. Assim, seja $CP_{i,j}$ um conjunto com todos os pontos que possuem posições candidatas em conflito com a posição candidata $x_{i,j}$, e $y_{i,j,c}$ uma variável de conflito entre a posição candidata $x_{i,j}$ e o ponto $c \in CP_{i,j} : c > i$. Com isso, o conjunto de restrições definido em 4.3 pode ser substituído, sem falta de generalização, pelo seguinte conjunto de restrições (Ribeiro e Lorena, 2004e,c):

$$|CP_{i,j}|\, x_{i,j} + \sum_{(k,t)\in S_{i,j}} x_{k,t} - \sum_{c\in CP_{i,j}} y_{i,j,c} \quad \leq \quad |CP_{i,j}| \qquad \begin{array}{l} \forall i = 1...NP \\ \forall j = 1...PC_i \end{array} \tag{4.10}$$

Como o conjunto descrito acima utiliza variáveis de sobreposições que indicam conflitos entre posições candidatas e pontos, a função objetivo também deve ser modificada, assim, o Problema de Minimização do Número de Conflitos baseado em pontos ($PMNC^p$), pode ser modelado como (Ribeiro e Lorena, 2004c):

$$v\left(PMNC^p\right) = Min \sum_{i=1}^{NP} \sum_{j=1}^{PC_i} \left(w_{i,j} x_{i,j} + \sum_{c\in CP_{i,j}} y_{i,j,c} \right) \tag{4.11}$$

Sujeito a:

$$\sum_{j=1}^{PC_i} x_{i,j} = 1 \qquad \forall i = 1...NP \tag{4.12}$$

$$|CP_{i,j}|\, x_{i,j} + \sum_{(k,t)\in S_{i,j}} x_{k,t} - \sum_{c\in CP_{i,j}} y_{i,j,c} \leq |CP_{i,j}| \qquad \begin{array}{l} \forall i = 1...NP \\ \forall j = 1...PC_i \end{array} \tag{4.13}$$

$$x_{i,j} \quad \text{e} \quad y_{i,j,c} \in \{0,1\} \qquad \begin{array}{l} \forall i = 1...NP \\ \forall j = 1...PC_i \\ c \in CP_{i,j} \end{array} \tag{4.14}$$

A Tabela 4.7 mostra os números médios de restrições geradas por cada uma das modelagens apresentadas. Os problemas utilizados foram os propostos por Yamamoto e Lorena (2005) e estão disponíveis em `www.lac.inpe.br/~lorena`. A primeira coluna mostra o número de pontos, seguidos pelo número de problemas, número médio de restrições para a formulação baseada em arestas e pelo número médio de restrições geradas pela formulação baseada em pontos. Percebe-se que esta última reduz de forma considerável o número de restrições.

TABELA 4.7 – Número médio de restrições geradas.

Número de pontos	**Número de problemas**	**Formulação baseada em arestas**	**Formulação baseada em pontos**
25	8	357	96
100	25	202	153
250	25	864	530
500	25	2909	1412
750	25	6181	2481
1000	25	10700	3643

4.4.1 LagClus para a modelagem baseada em pontos

A LagClus foi aplicada no $PMNC^p$, entretanto, como esta formulação é baseada em pontos e não em arestas, o particionamento foi feito de maneira diferente. Considere a Figura 4.8. Em (a) é mostrado o problema e em (b) o grafo obtido, assim como na Figura 4.7 (a) e (b).

Em um grafo G, uma clique pode ser definida com um subgrafo completo de G (Harary, 1972). Com isso, observando que as posições candidatas de cada ponto formam uma clique, foi aplicada a técnica de contração de vértices que consiste em agrupar todas as posições candidatas do ponto i, formando um único vértice. O grafo obtido então é particionado, nesse caso em dois, como mostra a Figura 4.8 (c). Em seguida, as contrações são expandidas obtendo o grafo original (d) , surgindo assim as arestas entre *clusters* (e),

e os subproblemas (f).

A contração dos vértices permite que nos subproblemas apareçam as restrições definidas em 4.12 (as cliques), e com isso, não precisam ser relaxadas. Observa-se na Figura 4.8 (e) que somente as arestas entre pontos são relaxadas.

Como as restrições definidas em 4.12 são respeitadas, durante o algoritmo de subgradientes somente a heurística HM é utilizada para melhorar uma solução relaxada dada pela LagClus.

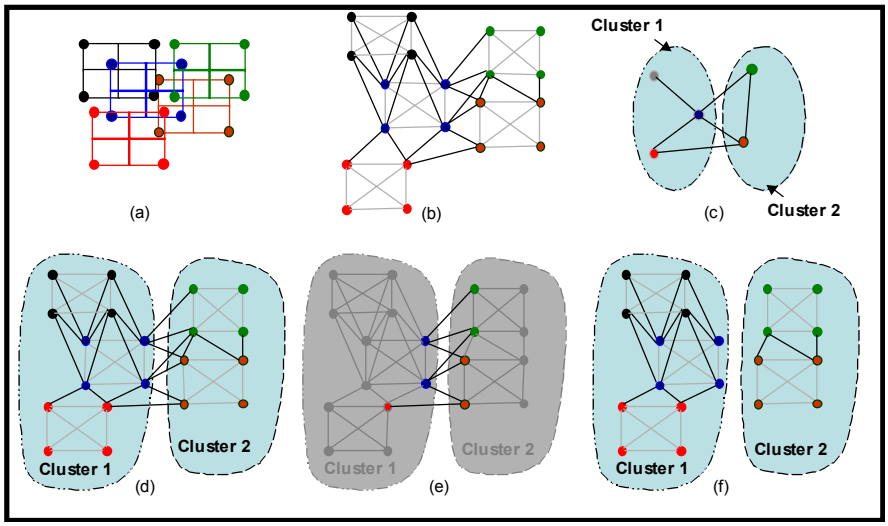


FIGURA 4.8 – Aplicação da LagClus para a formulação baseada em pontos.

TABELA 4.8 – Resultados médios obtidos com o CPLEX 7.5 (ILOG, 2001).

Problema	Limite inferior	Limite superior	GAP(%)	Tempo(s)	Rótulos em conflitos	Percentual de rótulos livres
25	27,75	27,75	0,00	0,20	4,88	80,50
100	100,00	100,00	0,00	0,03	0,00	100,00
250	250,00	250,00	0,00	0,06	0,00	100,00
500	500,84	500,84	0,00	0,74	1,68	99,67
750	757,25	759,12	0,25	9625,92	17,84	97,62
1000	1010,37	1051,92	3,94	6683,80	97,12	90,29

4.4.2 Resultados computacionais da LagClus para a modelagem baseada em pontos

A Tabela 4.8 mostra os resultados médios obtidos com o CPLEX 7.5 utilizando a modelagem baseada em pontos 4.11-4.14, para as instâncias testes. O CPLEX foi executado até resolver a instância ou parar por falta de memória. A primeira coluna

exibe o número de pontos a serem rotulados, seguida pelo limite inferior e superior, e pelo $GAP = \frac{(LimiteSuperior - LimiteInferior)}{LimiteSuperior} * 100$ que fornece, em porcentagem, o *Gap* encontrado entre os dois limitantes. Na quinta coluna são mostrados os tempos computacionais utilizados, seguido pelo número de rótulos em conflito e pelo percentual de rótulos livres para cada problema.

Para os problemas com até 500 pontos, o CPLEX resolveu todos em um tempo computacional muito baixo (menor que 0,74s), entretanto para problemas maiores, o CPLEX parou muitas vezes por falta de memória. Para problemas com 750 pontos, em 12 das 25 instâncias, ele encontrou a optimalidade. Já para 1000 pontos, nenhuma solução ótima foi encontrada.

A Tabela 4.9 apresenta os resultados médios obtidos com a aplicação da LagClus. Nesta tabela, além dos campos mostrados na Tabela 4.8, está o número de clusters usado em cada problema. Novamente o METIS foi usado para realizar o particionamento. Pode-se verificar facilmente que os *Gaps* de dualidade encontrados para as instâncias maiores, são menores que os obtidos pelo CPLEX. Para os problemas maiores (750 e 1000 pontos), a qualidade das soluções obtidas (ver Percentual de rótulos livres) foi superior às obtidas pelo CPLEX, em um tempo computacional menor.

TABELA 4.9 – Resultados médios obtidos com a LagClus para $PMNC^p$.

Problema	**Número de *clusters***	**Limite inferior**	**Limite superior**	**GAP(%)**	**Tempo(s)**	**Rótulos em conflitos**	**Percentual de rótulos livres**
25	2	25,62	28,13	8,67	3,50	6,00	76,00
100	2	100,00	100,00	0,00	0,12	0,00	100,00
250	2	250,00	250,00	0,00	0,12	0,00	100,00
500	2	500,84	500,84	0,00	0,40	1,68	99,67
750	10	758,09	758,96	0,12	53,84	17,60	97,65
1000	25	1030,07	1047,32	1,64	3445,40	90,16	90,98

Para as instâncias com 750 pontos, a LagClus encontrou a optimalidade em 21 das 25 instâncias. Para problemas com 1000 pontos, assim como aconteceu com o CPLEX, nenhuma solução ótima foi encontrada, no entanto o *Gap* de dualidade foi duas vezes menor.

Para mostrar a relação entre particionamento e qualidade da solução, foram feitos mais dois experimentos com as instâncias de 1000 pontos variando o número de *clusters* para 20 e 30 *clusters*. Os resultados estão mostrados na Tabela 4.10. Percebe-se claramente que, com a redução do número de *clusters* as soluções obtidas ficam melhores (ver *Gap* e Percentual de rótulos livres) no entanto, o tempo computacional aumenta. Por outro

lado, ao aumentar o número de *clusters* para 30, as soluções encontradas foram piores que as obtidas com 20 e 25 *clusters*, porém o tempo computacional reduz.

TABELA 4.10 – Resultados médios obtidos com a LagClus para o $PMNC^p$ sobre as instâncias com 1000 pontos variando o número de *clusters*.

Número de *clusters*	**Limite inferior**	**Limite superior**	**GAP(%)**	**Tempo(s)**	**Rótulos em conflitos**	**Percentual de rótulos livres**
20	1031,23	1044,80	1,30	3842,84	85,80	91,42
25	1030,07	1047,32	1,64	3445,40	90,16	90,98
30	1026,81	1049,16	2,13	734,80	93,56	90,64

4.5 Comparação dos resultados encontrados com os da literatura

A Tabela 4.11 compara os resultados obtidos com a LagClus para o $PMNC^a$ e para o $PMNC^p$, com os resultados relatados na literatura. Como pode ser verificado, os resultados encontrados com a LagClus sobre a modelagem baseada em pontos ($PMNC^p$) forneceu melhores resultados que a literatura. Nesta tabela existem vários algoritmos com objetivos distintos, como mostrou a Seção 4.2. Mesmo a formulação de minimização do número de conflitos sendo um limitante inferior para as demais abordagens, a mesma com a LagClus, apresentou melhores resultados.

TABELA 4.11 – Comparação com outros algoritmos.

Algoritmo	**Percentual de rótulos livres**				
	Problemas				
	100	250	500	750	1000
LagClus para $PMNC^p$	100,00	100,00	99,67	97,65	91,42 / 90,98 /90,64
$PMNC^p$ – CPLEX 7.5	100,00	100,00	99,67	97,62	90,29
LagClus para $PMNC^a$	100,00	100,00	99,38	95,53	86,47
$PMNC^a$ – CPLEX 7.5	100,00	100,00	99,67	NC	NC
AGC Melhor (Yamamoto e Lorena, 2005)	100,00	100,00	99,60	97,10	90,70
AGC Média (Yamamoto e Lorena, 2005)	100,00	100,00	99,60	96,80	90,40
Busca Tabu (Yamamoto et al., 2002)	100,00	100,00	99,30	96,80	90,00
AG com máscara (Verner et al., 1997)	100,00	99,98	98,79	95,99	88,96
AG sem máscara (Verner et al., 1997)	100,00	98,40	92,59	82,38	65,70
Simulated Annealing (Christensen et al., 1995)	100,00	99,90	98,30	92,30	82,09
Zoraster (1990)	100,00	99,79	96,21	79,78	53,06
Hirsh (1982)	100,00	99,58	95,70	82,04	60,24
3-opt Gradient Descent (Christensen et al., 1995)	100,00	99,76	97,34	89,44	77,83
2-opt Gradient Descent (Christensen et al., 1995)	100,00	99,36	95,62	85,60	73,37
Gradient Descent (Christensen et al., 1995)	98,64	95,47	86,46	72,40	58,29
Greedy Algorithm (Christensen et al., 1995)	95,12	88,82	75,15	58,57	43,41

4.6 Considerações finais

Este capítulo apresentou os resultados obtidos com a LagClus para o problema da rotulação cartográfica de pontos. Como observado, as soluções encontradas foram

significativas, principalmente para a formulação baseada em pontos, que reduziu o número de restrições geradas pela formulação baseada em arestas.

A estratégia de não relaxar as cliques formadas pelas posições candidatas de cada ponto, mostrou-se bem adaptada para este problema, dado que a LagClus produziu resultados melhores que os apresentados na literatura.

Baseado na idéia da redução do número de restrições explorando as cliques, no próximo capítulo é apresentado o problema da programação diária de fotos para um satélite de observação, junto com uma formulação reduzida. O objetivo é representar este problema através de um grafo de conflitos, e aplicar a LagClus neste problema, aproveitando as cliques geradas pelo modelo reduzido. Espera-se com isso, obter bons limitantes também para este problema.

CAPÍTULO 5

PROBLEMA DA PROGRAMAÇÃO DIÁRIA DE FOTOS DE UM SATÉLITE DE OBSERVAÇÃO

O Problema da Programação Diária de Fotos (PPDF) de um satélite de observação da Terra, é um dos problemas chave para um satélite de observação (Vasquez e Hao, 2003). O objetivo deste problema é programar um subconjunto de fotos de um conjunto maior denominado conjunto de fotos candidatas, que deverão ser fotografadas pelo satélite. Entretanto, existe um número grande de restrições (maior parte de adjacências ou conflitos) que restringem determinadas situações.

A função objetivo deste problema reflete vários critérios como a importância do cliente, demanda urgente, previsões meteorológicas, etc. As restrições incluem restrições físicas tal como a capacidade de armazenamento do satélite; e restrições lógicas como a não sobreposição de fotos e um tempo mínimo de transição entre duas fotos sucessivas.

Existem na literatura vários métodos para resolver este problema. Verfaillie et al. (1996) apresentaram um algoritmo exato denominado *Pseudo Dynamic Search*, que inclui um *Branch-and-Bound* dentro de um processo iterativo de solução. Esse método foi capaz de resolver 14 instâncias testes que foram mais tarde apresentadas formalmente por Bensana et al. (1999). O método resolveu as 13 instâncias consideradas simples, e das 7 consideradas difíceis, o método resolveu apenas 1.

Dentre os resultados mais interessantes, estão os de Vasquez e Hao (2001) que apresentam uma Busca Tabu para esse problema. A Busca Tabu permitiu encontrar as soluções ótimas de todas as instâncias simples, e nas instâncias difíceis, melhorou as soluções conhecidas.

Vasquez e Hao (2001) também apresentam uma formulação matemática para o PPDF. Mais tarde, Vasquez e Hao (2003) apresentaram bons limitantes superiores para este problema, o que permitiu avaliar os resultados obtidos em 2001 com uso da Busca Tabu.

Os limitantes foram obtidos a partir de uma representação em grafo de conflitos do problema. Em seguida, os autores particionaram o grafo obtido, formando *clusters*. Com isso, as arestas que conectam os *clusters* foram removidas e os subproblemas resolvidos separadamente. Ao final, os resultados das funções objetivos de cada *cluster* são somados, gerando um limitante superior. Vasquez e Hao (2003) procuraram reduzir ao máximo o número de arestas entre *clusters*, com isso os limitantes obtidos foram de boa qualidade. Por outro lado, os tempos computacionais foram altos, variando de horas a dias em um

Pentium III com um *solver* implementado pelos autores.

Devido à estratégia de particionamento adotada pelos autores para obter bons limitantes superiores, pode-se perceber que existe uma ligação entre essa técnica e a LagClus.

5.1 Formulação matemática

Seja $F = \{f_1, f_1, ..., f_n\}$ um conjunto de fotografias candidatas do tipo *mono* e *stereo*, que devem ser programadas para o próximo dia de observação de um satélite.

Cada fotografia tem associado:

- um benefício, que representa uma série de critérios como a importância do cliente, demanda de urgência, previsões, dentre outros fatores;
- um tamanho, que representa o tamanho necessário de memória para armazenar a foto no momento em que a mesma é tomada;
- um conjunto de possibilidades que corresponde aos diferentes modos de tirar a fotografia f_i: (1) para *mono*, que indica existirem 3 possibilidades de tirar a foto ou seja, pode ser usada qualquer uma das três câmeras do satélite (x, y ou z); ou (2) para *stereo*, limita o número de possibilidades em uma porque uma foto *stereo* necessita, simultaneamente, da câmera da frente e da que está atrás.

No PPDF existem três conjuntos de restrições:

- Restrições binárias. Para alguns pares (foto, câmera), é proibido programar simultaneamente a foto f_1 na câmera x e a foto f_2 na câmera y;
- Restrições ternárias; Para alguns pares (foto, câmera), é proibido programar simultaneamente a foto f_1 na câmera x, f_2 na câmera y e f_3 na câmera z;
- Restrição de capacidade ou da mochila. A soma dos tamanhos das fotos programadas não pode exceder a capacidade de gravação disponível no satélite.

Com isso, o objetivo do PPDF é encontrar um subconjunto F' de F que satisfaça todas as restrições definidas anteriormente, e que maximize a soma dos benefícios de cada fotografia em F'.

Seja F o conjunto de fotografias candidatas e $n = |F|$. Cada fotografia *mono* está associada a 3 pares de elementos: $(f_i, camera_1)$, $(f_i, camera_2)$ e $(f_i, camera_3)$. De modo similar, cada foto do tipo *stereo* esta associada a um par de elementos representado por $(f_i, camera_{13})$. Sendo n_1 e n_2 os números, respectivamente, de fotos *mono* e *stereo* em F $(n = n_1 + n_2)$, existe um total $m = 3n_1 + n_2$ de pares possíveis de elementos. Agora, pode-se definir uma variável de decisão binária x_i para cada um desses pares de elementos. Se $x_i = 1$, o correspondente par (foto, câmera) está presente na programação, caso contrário $x_i = 0$.

Seja g_i e c_i variáveis que representam, respectivamente, o benefício e o tamanho associado à variável x_i. Essas duas variáveis serão utilizadas para descrever a função objetivo e a restrição da mochila presentes na formulação.

Considerando as definições acima, o problema da programação diária de fotos pode ser formulado como (Vasquez e Hao, 2001):

$$v\,(PPDF) = Max\left\{\sum_{i=1}^{m} g_i x_i\right\} \tag{5.1}$$

Sujeito a:

$$\sum_{i=1}^{m} c_i x_i \leq CM \tag{5.2}$$

$$x_i + x_j \leq 1 \qquad \forall\,(x_i, x_j) \in C_2 \tag{5.3}$$

$$x_i + x_j + x_k \leq 2 \qquad \forall\,(x_i, x_j, x_k) \in C_{31} \tag{5.4}$$

$$x_i + x_j + x_k \leq 1 \qquad \forall\,(x_i, x_j, x_k) \in C_{32} \tag{5.5}$$

$$x_i \in \{0, 1\} \qquad i \in \{1, \ldots, m\} \tag{5.6}$$

Onde:

- CM é a capacidade máxima de armazenamento do satélite;
- C_2 é um conjunto de restrições que evita a sobreposição de fotos, respeita o tempo mínimo entre duas fotos sucessivas na mesma câmera, e impõe limitações no fluxo de dados instantâneo;
- C_{31} é um conjunto de restrições que não podem ser expressas usando restrições de sobreposição;
- C_{32} é um conjunto de restrições associadas às fotos *mono* que evita que uma mesma foto *mono* seja tomada mais de uma vez.

A restrição 5.2 é uma restrição da mochila. Já a restrição definida em 5.3 representa as restrições de conflito entre duas variáveis, ou seja, restrições de adjacências. Em 5.4 estão apresentadas restrições ternárias (três variáveis) que não podem ser representadas usando 5.3. As restrições definidas em 5.5 não permitem que uma foto *mono* seja tomada mais de uma vez. Usando conceitos de grafo, cada restrição gerada em 5.5, forma uma clique ou um conjunto de restrições de adjacências. E por último, a restrição 5.6 garante que todas as variáveis do problema são binárias.

5.2 Redução usando cliques maximais que contém um vértice e cobertura de vértices

Murray e Church (1997a) trabalhando com o *PMCIV* propuseram uma nova forma de representar restrições de adjacências, que permite reduzir o número de restrições de um modelo. Eles exploram o conceito de cliques maximais e de cobertura de vértices. Uma clique maximal é definida por Murray e Church (1997a) como sendo a maior clique que envolve um determinado vértice. Por exemplo, considere a Figura 5.1 que apresenta um grafo de conflito. A maior clique que envolve o vértice x_1 é dado pelo conjunto de vértices x_1, x_2, x_3 e x_4.

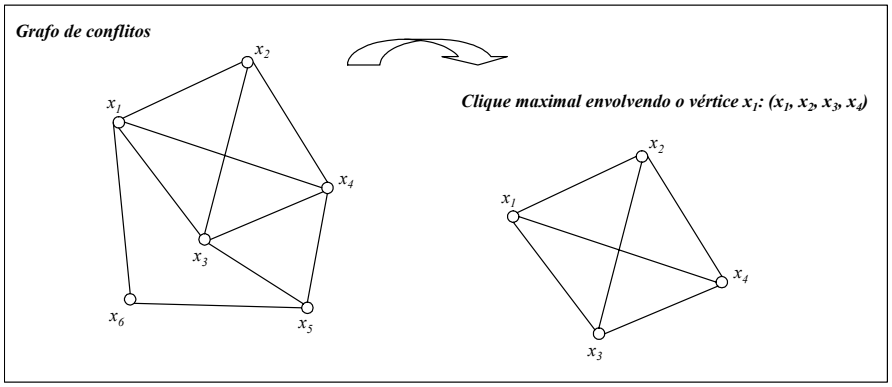


FIGURA 5.1 – Clique maximal envolvendo o vértice x_1.

Entretanto, um conjunto de restrições baseadas em cliques maximais podem conter restrições (cliques) redundantes. Isso significa que uma clique maximal de um vértice pode ser um subconjunto de uma clique maximal de outro vértice. Com isso, faz-se necessário identificar somente cliques maximais únicas para evitar redundância. Esse processo de identificação deve ser feito comparando-se as novas cliques maximais obtidas com as que foram identificadas previamente. Com isso, pode-se dizer que o número de cliques maximais únicas é limitado pelo número de vértices presentes no grafo de conflitos (Murray e Church, 1997a).

Porém, ao selecionar o conjunto de cliques maximais, não existe garantia de que todas as restrições de adjacências estejam satisfeitas. Considerando novamente a Figura 5.1, a aresta (x_1, x_6) não faz parte da clique maximal do vértice x_1, mostrada anteriormente. Com isso, faz-se necessário adicionar às restrições de cliques maximais, restrições de cobertura de vértices ou de empacotamento de vértices. Assim, são adicionadas, no máximo, um número de restrições de cobertura igual ao número de vértices do grafo de conflitos (Murray e Church, 1997a).

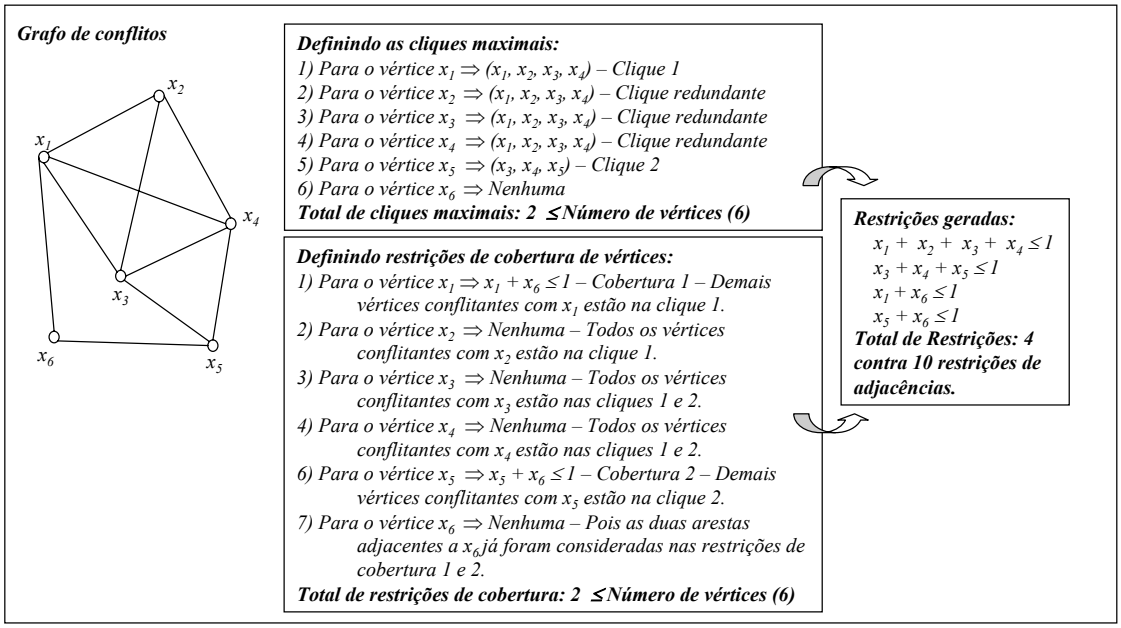


FIGURA 5.2 – Exemplo de aplicação da redução de Murray e Church (1997a).

A Figura 5.2 apresenta a redução proposta por Murray e Church (1997a) aplicada ao grafo mostrado na Figura 5.1. Observe que o número de restrições obtido ao final foi igual 4 que corresponde a 2 restrições de cliques maximais envolvendo os vértices x_1 e x_5, e 2 restrições de cobertura envolvendo os vértices x_1 e x_5. Observe ainda que este número de restrições é inferior a 2 vezes o números de vértices conforme Murray e Church (1997a).

5.3 Definição das restrições de redução

Seja CL_k o conjunto de vértices pertencentes à clique maximal k, para $k = \{1, ..., K\}$, tal que K representa o número total de cliques maximais. Assim, as restrições de cliques maximais podem ser expressas como:

$$\sum_{j \in CL_k} x_j \leq 1 \qquad \forall k = \{1, ..., K\} \tag{5.7}$$

Seja N_i um conjunto com todos os vértices que compartilham uma aresta com o vértice i, e n_i um coeficiente igual ao maior número de elementos do conjunto N_i que podem ser selecionados simultaneamente sem violar uma restrição de adjacência. Sendo assim, as restrições de cobertura de vértices podem ser definidas como:

$$n_i x_i + \sum_{j \in N_i} x_j \leq n_i \qquad \forall i = \{1, ..., V\} \tag{5.8}$$

Sendo V o conjunto de vértices do grafo de conflitos.

Porém dessa forma, as restrições definidas em 5.7 aparecem também nas restrições de vértices definidas em 5.8, indicando redundância de restrições. Para evitar isso, seja $\hat{N}_i = \{j \in N_i | i, j \notin CL_k \forall k = 1, ..., K\}$ o conjunto dos vértices j que compartilham uma aresta com i, dado que a aresta (i, j) não aparece em nenhuma clique maximal k, e $\hat{n}_i$ um coeficiente igual ao maior número de elementos do conjunto $\hat{N}_i$ que podem ser selecionados simultaneamente. Sendo assim, a restrição definida em 5.8 pode ser substituída por:

$$\hat{n}_i x_i + \sum_{j \in \hat{N}_i} x_j \leq \hat{n}_i \qquad \forall i = \{1, ..., V\} : \hat{N}_i \neq \emptyset \tag{5.9}$$

Com isso, as restrições definidas em 5.7 e em 5.9 podem substituir, sem perda de generalidade, o conjunto de restrições de adjacências presentes em um grafo de conflitos.

5.4 Algoritmo de redução

O processo de geração das restrições definidas em 5.7 e 5.9 está apresentado na Figura 5.3.

Este processo é constituído de duas fases, diferencidas pelos retângulos maiores. O primeiro passo é identificar as cliques maximais. Isso é feito para cada vértice i e consiste

em uma procura no conjunto N_i do maior subconjunto de vértices (incluindo i) que é mutuamente conectado. O segundo passo consiste no processo de criação das restrições de cobertura de vértices que constitue das restrições de adjacências não consideradas no passo anterior, e na definição do coeficiente $\hat{n}_i$.

Ao término do algoritmo mostrado na Figura 5.3, as restrições obtidas podem substituir todas as restrições de adjacências presentes em um grafo de conflitos.

Na próxima seção, será apresentada uma formulação reduzida para o PPDF que utiliza os conceitos apresentados nesta seção.

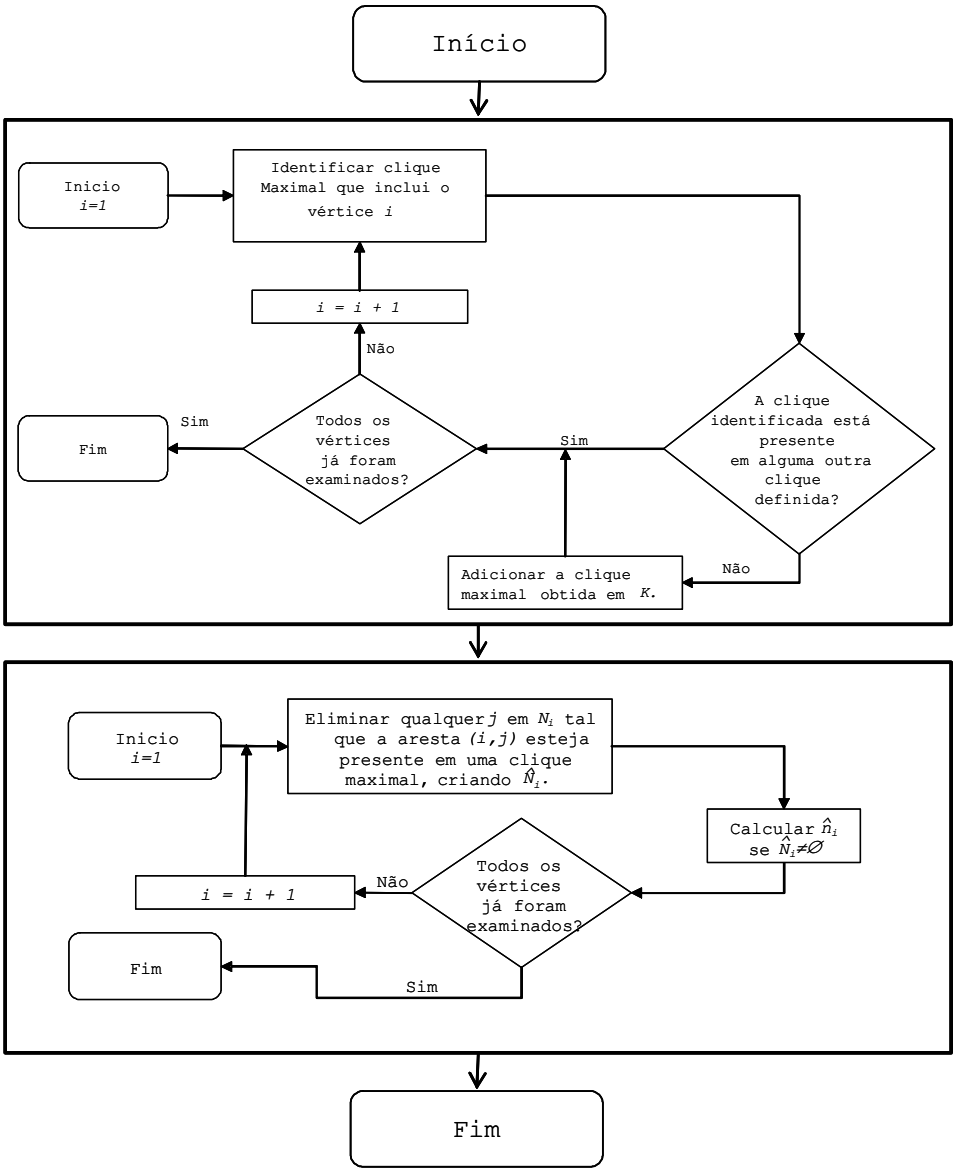


FIGURA 5.3 – Processo de geração das restrições de Murray e Church (1997a).

5.5 Formulação reduzida para o PPDF

As restrições definidas em 5.3 e 5.5, podem ser reduzidas através da formulação de Murray e Church (1997a). Com isso, o problema original do PPDF pode ser reescrito como:

$$v\left(PPDF\right) = Max\left\{\sum_{i=1}^{m} g_i x_i\right\} \tag{5.10}$$

Sujeito a:

$$\sum_{i=1}^{m} c_i x_i \leq CM \tag{5.11}$$

$$\sum_{j \in CL_k} x_j \leq 1 \qquad \forall k \in K \tag{5.12}$$

$$\hat{n}_i x_i + \sum_{j \in \hat{N}_i} x_j \leq \hat{n}_i \qquad \forall i = \{1, ..., V\} : \hat{N}_i \neq \emptyset \tag{5.13}$$

$$x_i + x_j + x_k \leq 2 \qquad \forall \left(x_i, x_j, k_k\right) \in C_{31} \tag{5.14}$$

$$x_i \in \{0, 1\} \qquad i \in \{1, \ldots, m\} \tag{5.15}$$

Na próxima seção são mostrados os resultados obtidos com a redução acima.

5.6 Resultados obtidos com a redução

A formulação apresentada em 5.1-5.6 foi aplicada nas instâncias testes propostas por Bensana et al. (1999), assim como a formulação apresentada em 5.10-5.15. Os resultados estão mostrados nas Tabelas 5.1 e 5.2.

As colunas nestas tabelas referem-se a:

- Inst. - Nome da instância teste;
- Tipo 1 - Número de restrições definidas em 5.3;
- Tipo 2 - Número de restrições definidas em 5.4;

- Tipo 3 - Número de restrições definidas em 5.5
- Total1 - Número total de restrições (Tipo 1 + Tipo 2 + Tipo3) gerada pela formulação de Vasquez e Hao (2001);
- Cliques maximais- Número de restrições definidas em 5.12;
- Clique máxima - Tamanho da maior clique encontrada;
- Restrições de nó- Número de restrições definidas em 5.13;
- Total2 - Número total de restrições (Cliques maximais + Restrições de nó + Tipo 2) gerada pela formulação reduzida;
- Dif - Diferença (Total1 - Total2) entre o número de restrições geradas pela formulação de Vasquez e Hao (2001) e a formulação reduzida.

TABELA 5.1 – Resultados do processo de redução para as instâncias sem a restrição da mochila

Inst.	**Modelo de Vasquez e Hao (2001)**				**Modelo reduzido**					**Dif.**
	Restrições				**Cliques**	**Clique**	**Restrições**	**Tipo 2**	**Total2**	
	Tipo 1	**Tipo 2**	**Tipo 3**	**Total1**	**maximais**	**máxima**	**de nó**			
54	389	23	67	479	100	8	46	23	169	310
29	610	0	82	692	99	9	48	0	147	545
42	1762	64	190	2016	268	14	157	64	489	1527
28	6302	590	230	7122	339	22	204	590	1133	5989
5	13982	367	309	14658	785	21	596	367	1748	12910
404	919	18	100	1037	142	15	60	18	220	817
408	2560	389	200	3149	317	16	159	389	865	2284
412	6585	389	300	7274	533	20	334	389	1256	6018
414	9456	4719	364	14539	668	23	441	4719	5828	8711
503	705	86	143	934	196	9	75	86	357	577
505	2666	526	240	3432	393	16	168	526	1087	2345
507	5545	2293	311	8149	550	18	327	2293	3170	4979
509	7968	3927	348	12243	627	22	400	3927	4954	7289

TABELA 5.2 – Resultados do processo de redução para as instâncias com a restrição da mochila.

Inst.	**Modelo de Vasquez e Hao (2001)**				**Modelo reduzido**					**Dif.**
	Restrições				**Cliques**	**Clique**	**Restrições**	**Tipo 3**	**Total2**	
	Tipo 1	**Tipo 2**	**Tipo 3**	**Total1**	**maximais**	**máxima**	**de nó**			
1401	11893	2913	488	15294	879	21	593	2913	4385	10909
1403	14997	3874	665	19536	1260	21	846	3874	5980	13556
1405	24366	4700	855	29921	1758	21	1294	4700	7752	27169
1407	30058	5875	1057	36990	2275	21	1656	5875	9806	27184
1502	296	29	209	534	149	5	146	29	324	210
1504	5106	882	605	6593	1012	12	498	882	2392	4201
1506	19033	4775	940	24748	1980	19	1365	4775	8120	16628

A Tabela 5.1 apresenta os resultados para as instâncias consideradas simples. Essas instâncias, além de serem menores, não apresentam a restrição da mochila. Pode-se perceber que a redução aplicada diminui consideravelmente o número de restrições. A maior diferença obtida foi na instância "5", enquanto a formulação original gera um total de 14658 restrições, a formulação reduzida apresenta 1748, uma diferença de 12910 restrições. A maior clique encontrada nesse caso contém 21 vértices.

A Tabela 5.2 apresenta os resultados para as instâncias consideradas difíceis que neste caso, apresentam a restrição da mochila. Observa-se mais uma vez que o número de restrições geradas pelo modelo reduzido é bem menor que as do modelo original de Vasquez e Hao (2001). A maior redução aconteceu na instância "1407", de 36990 para 9806, uma redução de 27184 restrições.

TABELA 5.3 – Resultados obtidos com o CPLEX 7.5 para o PPDF.

Inst.	**Modelo de Vasquez e Hao (2001)**		**Modelo reduzido**	
	Solução	**Tempo (s)**	**Solução**	**Tempo (s)**
54	70	0,18	70	0,08
29	12032	0,08	12032	0,07
42	108067	0,47	108067	0,18
28	56053	4,61	56053	1,03
5	115	9,86	115	11,42
404	49	0,07	49	0,06
408	3082	0,81	3082	0,61
412	16102	1,24	16102	1,40
414	22120	4,80	22120	6,83
503	9096	0,07	9096	0,09
505	13100	0,24	13100	0,45
507	15137	2,18	15137	5,13
509	19125	8,34	19125	6,17
1502	61158	0,07	61158	0,06

A Tabela 5.3 apresenta os resultados obtidos com o CPLEX 7.5 para a modelagem de Vasquez e Hao (2001) e a reduzida, aplicadas aos problemas testes das Tabelas 5.1 e 5.2. Para os problemas mais difíceis, as duas formulações apresentaram problemas na definição da solução ótima, não fechando o *Gap* de dualidade em 10 horas de execução, com exceção da instância "1502", que foi resolvida por ambas as modelagens.

Em média, as instâncias foram resolvidas em 2,36s e 2,40s com a modelagem de Vasquez e Hao (2001) e a modelagem reduzida, respectivamente. A diferença de tempo é muito pequena, apesar da média apresentada pela modelagem reduzida ser superior a primeira. Uma grande diferença poderia ser obtida nos problemas mais difíceis, no entanto, as duas modelagens apresentaram problemas para encontrar a solução ótima.

5.7 Considerações finais

Este capítulo apresentou o problema da programação diária de fotos de um satélite de observação. Foi apresentada a modelagem matemática do problema proposta por Vasquez e Hao (2001) assim como, uma modelagem reduzida que utiliza os conceitos de cliques maximais e de cobertura de vértices.

A modelagem reduzida permite trabalhar com formulações menores, no entanto, os resultados ótimos para as instâncias difíceis não puderam ser obtidos. Por outro lado, a estratégia de particionamento utilizada por Vasquez e Hao (2003) para obter um limitante superior, se assemelha bem com a LagClus. Assim, pretende-se formar um grafo de conflitos como proposto por Vasquez e Hao (2003), e aplicar a LagClus nesse problema, porém considerando a modelagem reduzida.

Espera-se obter limitantes tão bons quanto ou até melhores que os obtidos por Vasquez e Hao (2003), em um tempo computacional menor.

CAPÍTULO 6

GERAÇÃO DE COLUNAS PARA O PROBLEMA DO CARREGAMENTO DE PALETES

O método ou algoritmo *Branch-and-Price* (Barnhart et al., 1998) consiste no uso combinado do método *Branch-and-Bound* com a técnica de Geração de Colunas, sobre um problema denominado Problema Mestre Restrito. Esse método tem sido aplicado em vários problemas como em problemas de localização (Senne et al., 2005), escalas de tripulações (Barnhart et al., 1998), dentre outros.

Entretanto, o *Branch-and-Price* (B&P) constitui a última fase de um algoritmo denominado Geração de Colunas (GC). A GC pode ser empregada quando todas as colunas de um problema não são conhecidas, e uma enumeração completa delas não é possível (Senne et al., 2005), ou quando o problema é obtido usando a decomposição Dantzig-Wolfe (Dantzig e Wolfe, 1960). Porém, não necessariamente faz-se necessário descer na árvore de busca, ou seja, deve-se aplicar o B&P. O B&P pode ser aplicado quando o algoritmo de Geração de Colunas termina e o vetor de solução não é inteiro (Wolsey, 1998).

O método de geração de colunas vem sendo utilizado em várias aplicações, como em problemas de roteamento de veículos (Desrochers et al., 1992), escala de tripulações (Barnhart et al., 1998; Mauri e Lorena, 2004) e de localização de facilidades (Senne e Lorena, 2001; Lorena e Senne, 2003; Senne et al., 2005). Muitas dessas aplicações utilizam técnicas de redução do número de colunas. Uma boa revisão sobre o assunto pode ser obtida no trabalho de Desaulniers et al. (2001).

Assim, o *Branch-and-Price* pode ser subdividido nas seguintes fases:

a) Estabelecer conjunto inicial de colunas e aplicar método de geração de colunas;

b) Descer em uma árvore de busca binária, adicionando colunas e realizando podas nos nós.

O primeiro item descrito anteriormente, está sendo estudado e os resultados estão mostrados nesta seção. A busca na árvore binária está em fase de desenvolvimento, constituindo no *Branch-and-Price*, objetivo desta proposta.

A seguir será apresentado o método de geração de colunas para o PMCIV usando a decomposição Dantzig-Wolfe proposta por Hicks et al. (2004) para o problema de máximo

conjunto independente de vértices com peso.

Na Seção 6.2 é apresentada a formulação do Problema de Carregamento de Paletes (PCP), seguida pela formulação e pelos resultados obtidos com a LagClus. Na Seção 6.5 são apresentados os resultados obtidos com a geração de colunas mostradas na Seção 6.1 para o PCP, e por último, na Seção 6.6, são apresentadas as considerações finais.

6.1 Geração de colunas para o PMCIV

A implementação clássica de geração de colunas, utiliza um problema mestre e subproblemas geradores de colunas. O problema mestre restrito (PMR), através de suas variáveis duais, direciona os subproblemas na busca por novas colunas que trazem informações novas para o PMR.

Assim, considere a formulação apresentada na Seção 3.2 para o PMCIV (por questões de facilidade, ela será reapresentada a seguir).

$$v\,(PMCIV) = Max \sum_{p=1}^{\overline{P}} x^p \tag{6.1}$$

Sujeito a:

$$+A^1x^1 \quad +A^2x^2 \quad \cdots \quad +A^{\overline{P}}x^{\overline{P}} \quad \leq \quad 1 \tag{6.2}$$

$$\begin{array}{rrrrcl} +D^1x^1 & & & & \leq & 1 \\ & +D^2x^2 & & & \leq & 1 \\ & & \cdots & & \leq & \vdots \\ & & & +D^{\overline{P}}x^{\overline{P}} & \leq & 1 \end{array} \tag{6.3}$$

$$x^1 \in B^{n_1} \quad \cdots \quad \cdots \quad x^{\overline{P}} \in B^{n_{\overline{P}}} \tag{6.4}$$

Sendo:

- x^p um vetor das variáveis de decisão do *cluster* p;
- A^p uma matriz binária de dimensões M x $|V|$ que representa os coeficientes das variáveis x^p do *cluster* p, presentes nas M restrições de adjacências entre *clusters*;

- D^p uma matriz binária de dimensões $|E| - M$ x $|V|$ que representa os coeficientes das variáveis x^p do *cluster* p, presentes nas restrições de adjacências intra *clusters*;

- B^{n_p} um vetor de variáveis binárias do *cluster* p de comprimento n_p.

Com isso, aplicando decomposição Dantzig-Wolfe para a relaxação de programação linear do problema 6.1-6.4 conforme Hicks et al. (2004), tem-se o seguinte PMR:

$$v(PMR)_{LP} = Max \left(\sum_{p=1}^{\overline{P}} \sum_{j \in J_p} \lambda_{jp} \bar{x}^{jp} \right) \tag{6.5}$$

Sujeito a:

$$\sum_{p=1}^{\overline{P}} \sum_{j \in J_p} \lambda_{jp} (A_p \bar{x}^{jp}) \leq 1 \tag{6.6}$$

$$\sum_{j \in J_p} \lambda_{jp} = 1 \qquad \forall p \in \left\{1...\overline{P}\right\} \tag{6.7}$$

$$\lambda_{jp} \geq 0 \qquad \forall p \in \left\{1...\overline{P}\right\}, j \in J_p \tag{6.8}$$

Sendo:

- J_p um conjunto de pontos extremos do *cluter* p;

- $\bar{x}^{jp}$ um vetor de comprimento $|V_p|$ que representa os pontos extremos $j \in J_p$;

- λ_{jp} uma variável de decisão que representa o ponto extremo $j \in J_p$.

Os subproblemas $p \in \left\{1, ..., \overline{P}\right\}$ são todos PMCIV definidos como:

$$v(PMCIV^p) = Max \left\{ \left(1 - A_p^T \alpha\right) \bar{x}^{jp} \right\} \tag{6.9}$$

Sujeito a:

$$D^p \bar{x}^{jp} \leq 1 \tag{6.10}$$

$$\bar{x}^{jp} \in B^{n_p} \tag{6.11}$$

Onde α é um vetor de comprimento M que representa as variáveis duais das restrições definidas em 6.6. Uma nova coluna, fornecida por um subproblema, será inserida no PMR, se o seu custo reduzido for positivo, ou seja, $v(PMCIV^p) - \beta_p > 0$, sendo β_p a variável dual associada com a p-ésima restrição definida em 6.7.

6.2 Problema do carregamento de paletes

O Problema do Carregamento de Paletes (PCP) consiste em: dado um conjunto de itens idênticos (caixas idênticas) com dimensões retangulares e um palete também retangular; procura-se otimizar a superfície do palete posicionando o máximo possível de caixas sobre o mesmo, sendo que as caixas não podem se sobrepor e todas elas devem estar posicionadas (empacotadas) sobre o palete, não podendo ultrapassar os limites de comprimento e largura do mesmo; elas ainda podem ser rotacionadas em 90º graus devendo estar com suas arestas paralelas às do palete. De acordo com Dyckhoff (1990) esse problema pode ser classificado como sendo bidimensional, com seleção de itens iguais sobre um objeto único, sendo assim, esse problema pertence a um caso especial de problemas de corte e empacotamento. A Figura 6.1 apresenta uma solução para o PCP. Observe que uma vez definida a primeira camada, as demais são idênticas, por isso esse problema pode ser reduzido ao caso bidimensional.

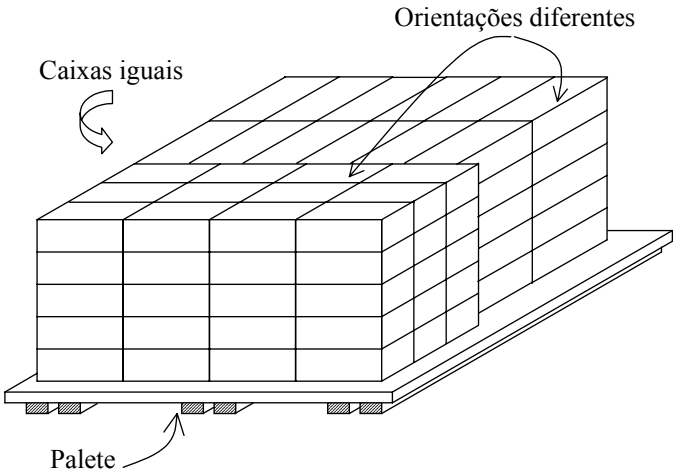


FIGURA 6.1 – Exemplo de uma solução para o PCP.

O PCP aparece freqüentemente na logística de distribuição de produtos e com isso, o número total de caixas empacotadas pode reduzir os custos logísticos dado que um

pequeno incremento nesse número, pode estar representando uma economia significativa.

Considerando as diversas aplicações práticas desse problema, muitos métodos de solução tem sido estudados. Os algoritmos exatos existentes utilizam basicamente uma estrutura em árvore (Dowsland, 1987; Bhattacharya et al., 1998; Alvarez-Valdes et al., 2004). Devido às dificuldades existentes, vários outros métodos foram criados ou utilizados, entre eles estão os métodos construtivos que dividem o palete em blocos (Young-Gun e Maing-Kyu, 2001), métodos recursivos (Young-Gun e Maing-Kyu, 1998) e métodos baseados em estruturas do tipo G4 (Scheithauer e Terno, 1996) e do tipo L (Lins et al., 2003). Existem outros trabalhos que aplicam as metaheurísticas conhecidas como Busca Tabu (Pureza e Morabito, 2005) e Algoritmos Genéticos (Herbert e Dowsland, 1996). Existem também vários limitantes superiores que consideram a geometria do problema, o que permite avaliar a qualidade de uma solução.

Esse problema analisado sob outro ponto de vista, é o clássico PMCIV (Dowsland, 1987). O PCP pode ser representado através de um grafo de conflitos em que cada vértice, indica a localização do canto inferior esquerdo de uma caixa posta na horizontal ou vertical, e as arestas do grafo, as possíveis sobreposições das caixas.

6.3 Formulação do problema de carregamento de paletes

O PCP pode ser formulado usando o caso particular da formulação de Beasley (1985) para o problema de corte não-guilhotinado bidimensional. Considerando a Figura 6.2, seja L e $W \in Z^+$, o comprimento e a largura do palete, respectivamente, tal que $L \geq W$, e, l e $w \in Z^+$, o comprimento e a largura das caixas, respectivamente, tal que $l \geq w$ e $l \leq Min\,(L, W)$. Para representar os possíveis modos de empacotar uma caixa, seja $(l_1, w_1) = (l, w)$ e $(l_2, w_2) = (w, l)$. Com isso, essas posições podem ser representadas por $(l_i, w_i)_{i=1,2}$, que indicam o comprimento e a largura de uma face na orientação i.

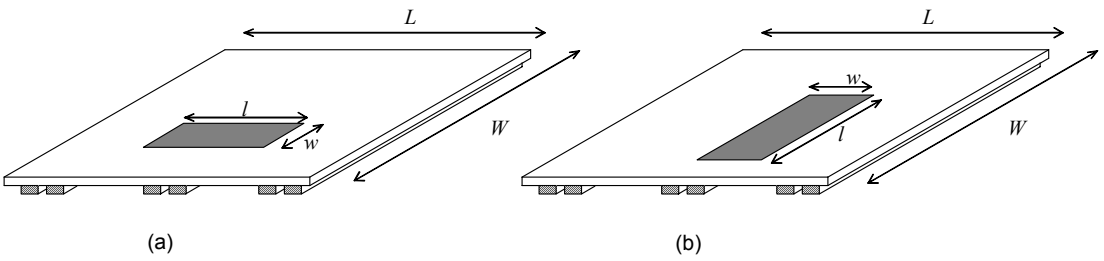


FIGURA 6.2 – Possíveis orientações para uma caixa na modelagem do PCP.

Para representar as posições das caixas no palete, seja X e Y dois conjuntos que juntos são utilizados para definir as coordenadas (p, q) do canto inferior esquerdo das caixas.

Esses conjuntos podem ser descritos como:

$$X = \left\{ p \in Z^{+} | p = \sum_{i=1}^{2} l_i b_i, 0 \leq p \leq L - w, b_i \geq 0, b_i \in Z^{+}, \forall i = 1, 2 \right\} \tag{6.12}$$

$$Y = \left\{ q \in Z^{+} | q = \sum_{i=1}^{2} w_i b_i, 0 \leq q \leq W - w, b_i \geq 0, b_i \in \mathrm{Z}^{+}, \forall i = 1, 2 \right\} \tag{6.13}$$

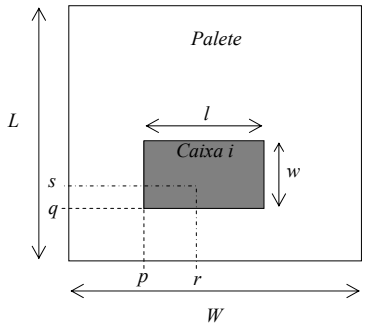


FIGURA 6.3 – Posição (r, s) não permitida em função da colocação de uma caixa na posição (p, q) com orientação i.

Seja a uma função que descreve as restrições de sobreposições no palete. Esta função pode ser obtida com antecedência para cada vértice (p, q) em relação a qualquer outro vértice (r, s), para cada orientação i, sendo $p \in X | p \leq L - l_i$, $q \in Y | q \leq W - w_i$, $r \in X$, $s \in Y$, e $i = 1, 2$ (Ver Figura 6.3). Assim, essa função pode ser expressa como:

$$a_{ipqrs} = \begin{cases} 1, & \text{Se: } 0 \leq p \leq r \leq p + l_i - 1 \leq L - 1,\ 0 \leq q \leq s \leq q + w_i - 1 \leq W - 1 \\ 0, & \text{caso contrário} \end{cases} \tag{6.14}$$

Agora, seja $x_{ipq} \in \{0, 1\}$ uma variável de decisão para todo $p \in X | p \leq L - l_i$, $q \in Y | q \leq W - w_i$, e $i = 1, 2$. Se $x_{ipq} = 1$, uma caixa é colocada nas coordenadas (p, q) do palete com a orientação i, caso contrário, $x_{ipq} = 0$.

Com isso, o PCP pode ser formulado como (Beasley, 1985):

$$v(PCP) = Max \left(\sum_{i=1}^{2} \sum\nolimits_{\{p \in X | p \leq L - l_i\}} \sum\nolimits_{\{q \in Y | q \leq W - w_i\}} x_{ipq} \right) \tag{6.15}$$

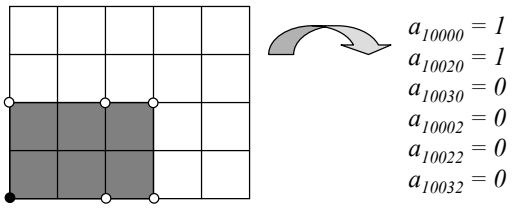
$$\sum_{i=1}^{2} \sum_{\{p \in X | p \leq L - l_i\}} \sum_{\{q \in Y | q \leq W - w_i\}} a_{ipqrs} x_{ipq} \leq 1, \forall r \in X, s \in Y \tag{6.16}$$

$$x_{ipq} \in \{0,1\} \, \forall i = 1...2, p \in X | p \leq L - l_i, q \in Y | q \leq W - w; \tag{6.17}$$

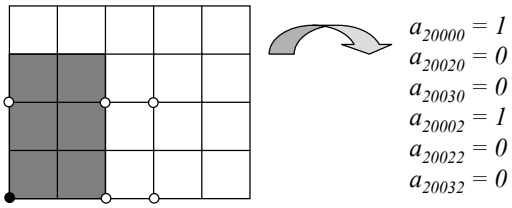
A restrição 6.16 garante a não existência de sobreposição de caixas e a 6.17 que as variáveis de decisão são binárias.

A seguir será apresentado um exemplo da formulação do PCP adaptado de Oliveira (2005). Considere que as dimensões do palete são $(L, W) = (5, 4)$ e as dimensões das caixas $(l, w) = (3, 2)$. As posições onde é possível colocar o canto inferior esquerdo das caixas são dadas pelos conjuntos $X = \{0, 2, 3\}$ e $Y = \{0, 2\}$ conforme as Equações 6.12 e 6.13, respectivamente. Definindo agora os valores da função a:

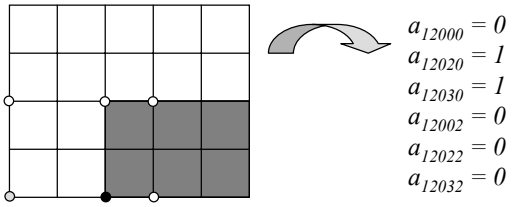
- Para $(p, q) = (0, 0)$ e $i = 1$:



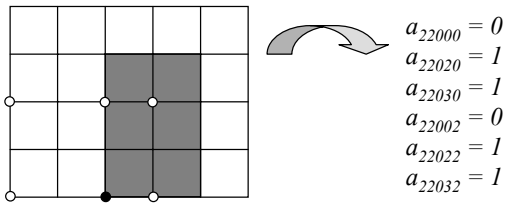
- Para $(p, q) = (0, 0)$ e $i = 2$:



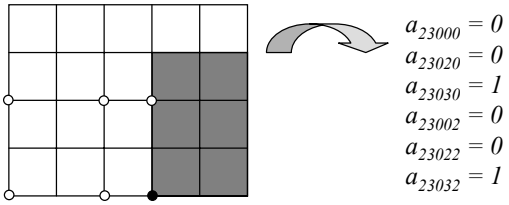
- Para $(p, q) = (2, 0)$ e $i = 1$:



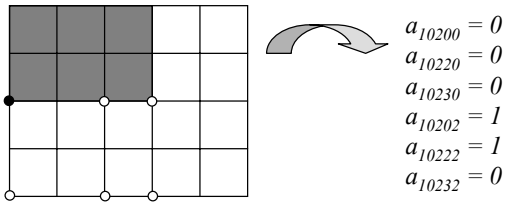
- Para $(p, q) = (2, 0)$ e $i = 2$:



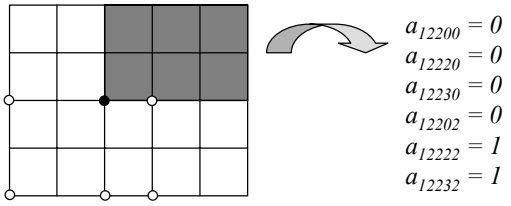
- Para $(p, q) = (3, 0)$ e $i = 2$:



- Para $(p, q) = (0, 2)$ e $i = 1$:



- Para $(p, q) = (2, 2)$ e $i = 1$:



Agora as restrições definidas em 6.16 podem ser escritas como:

- Para $(r, s) = (0, 0)$: $x_{100} + x_{200} \leq 1$;
- Para $(r, s) = (0, 2)$: $x_{102} + x_{200} \leq 1$;
- Para $(r, s) = (2, 0)$: $x_{100} + x_{120} + x_{220} \leq 1$;
- Para $(r, s) = (2, 2)$: $x_{102} + x_{122} + x_{220} \leq 1$;
- Para $(r, s) = (3, 0)$: $x_{120} + x_{220} + x_{230} \leq 1$;
- Para $(r, s) = (3, 2)$: $x_{122} + x_{220} + x_{230} \leq 1$.

Assim, o problema com $(L, W) = (5, 4)$ e $(l, w) = (3, 2)$ pode ser formulado da seguinte maneira:

$$v\,(PCP) = Max\,(x_{100} + x_{102} + x_{120} + x_{122} + x_{200} + x_{220} + x_{230})$$

Sujeito a:

$$x_{100} + x_{200} \leq 1$$

$$x_{102} + x_{200} \leq 1$$

$$x_{100} + x_{120} + x_{220} \leq 1$$

$$x_{102} + x_{122} + x_{220} \leq 1$$

$$x_{120} + x_{220} + x_{230} \leq 1$$

$$x_{122} + x_{220} + x_{230} \leq 1$$

$$x_{100}, x_{102}, x_{120}, x_{122}, x_{200}, x_{220}, x_{230} \in \{0, 1\}$$

Para o modelo acima, a solução ótima é dada por $x_{100} = x_{102} = x_{230} = 1$ e $x_{120} = x_{122} = x_{200} = x_{220} = 0$ com o valor da função objetivo igual a 3. A Figura 6.4 mostra graficamente a distribuição ótima das caixas sobre o palete.

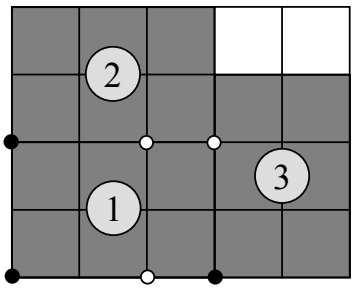


FIGURA 6.4 – Solução ótima para o problema $(L, W) = (5, 4)$ e $(l, w) = (3, 2)$.

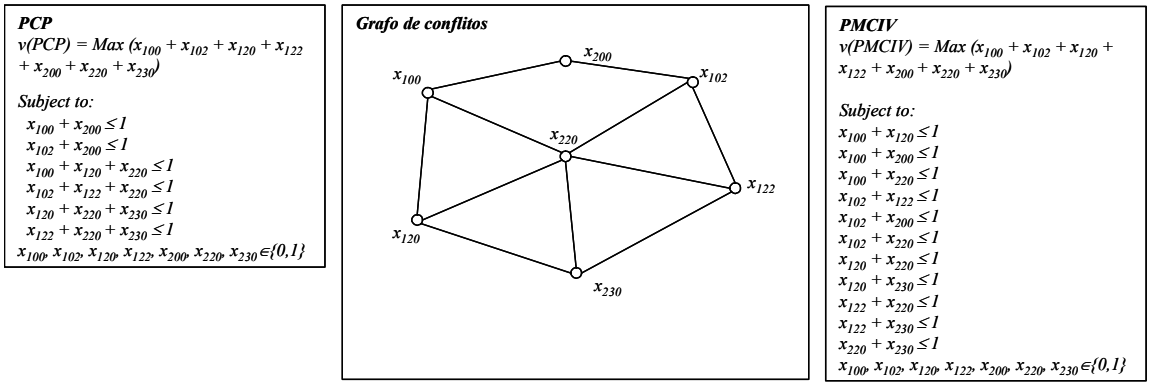


FIGURA 6.5 – Comparação entre o PCP e o PMCIV.

A formulação mostrada no Capítulo 2 para o PMCIV, produz mais restrições que a formulação definida em 6.15- 6.17. Isso decorre do fato de que a formulação de Beasley (1985) explora o uso das cliques, reduzindo assim o número de restrições. Por exemplo, considere novamente o problema apresentado acima no qual as dimensões do palete são $(L, W) = (5, 4)$ e as das caixas $(l, w) = (3, 2)$. A Figura 6.5(a) mostra a formulação produzida pelo modelo 6.15-6.17. Usando a abordagem dada pelo PMCIV, a Figura 6.5(b) mostra o grafo de conflitos obtido para o problema, e em (c) é apresentada a modelagem do PMCIV. Como previsto antes, o PMCIV gera mais restrições que a modelagem de Beasley (1985), entretanto essas restrições estão consideradas implicitamente na formulação do PCP.

6.4 Aplicação da LagClus e resultados

A LagClus para o PCP pode ser aplicada sobre a formulação de Beasley (1985), realizando os seguintes passos:

a) Montar o grafo de conflitos G a partir da modelagem do PCP e aplicar uma heurística de particionamento no grafo G, dividindo-o em $\overline{P}$ *clusters*;

b) Relaxar as cliques presentes no PCP que apresentam vértices em mais de um *cluster*. Em cada clique relaxada, analisar se existem pares de vértices que pertencem a um mesmo *cluster*, se existir, adicionar ao respectivo *cluster* uma restrição de adjacência entre cada par encontrado;

c) A relaxação lagrangeana obtida então é dividida em $\overline{P}$ subproblemas e resolvida.

Deve-se observar que no passo b) se alguma clique for relaxada, esta deve ser decomposta e cada uma de suas arestas analisada. Se uma aresta conecta dois vértices de um mesmo *cluster*, a mesma deve ser acrescentada no respectivo *cluster*. Esse procedimento fortalece a relaxação lagrangeana. Novamente, o particionamento foi feito usando o METIS.

Uma solução relaxada pode não ser uma solução factível, assim, no algoritmo de subgradientes é utilizada uma heurística de verificação e melhoria (HVM). Essa heurística primeiramente identifica todos os vértices da solução relaxada que estão em conflitos. Em seguida, remove da solução relaxada o vértices com o maior número de conflitos. O processo é novamente repetido, até obter uma solução totalmente sem conflitos, ou seja, uma solução factível. Por fim, a heurística testa a inserção dos outros vértices de modo a maximizar o número de vértices na solução factível. Esse algoritmo está representado na Figura 6.6.

As Tabelas 6.1, 6.2 e 6.3 apresentam os resultados obtidos com a LagClus para três conjuntos de instâncias testes. O número de *clusters* utilizado em cada conjunto foi obtido através de experimentos, procurando encontrar um número capaz de fornecer um limitante satisfatório com baixo custo computacional.

Heurística de Verificação e Melhoria - HVM
1. Fazer o vetor de solução factível (v_f) igual ao vetor da solução relaxada dado pela LagClus;
2. Enquanto não obtiver uma solução factível
   3. Para cada vértice i de v_f, definir o número de vértices j conflitantes com i;
   4. Ordenar, decrescentemente, v_f conforme o número de vértices conflitantes;
   5. Eliminar o primeiro vértice presente em v_f;
6. Fim do enquanto;
7. Testar dentre os demais vértices não presentes em v_f, se é possível inserir mais algum em v_f.

FIGURA 6.6 – Heurística de Verificação e Melhoria (HVM) aplicada no algoritmo de subgradientes para o PCP

Nas tableas as colunas representam:

- Instância - Nome dado a instância;
- L e W - Comprimento e largura do palete, respectivamente;
- l e w - Comprimento e largura das caixas, respectivamente;
- Solução Ótima - Solução ótima do problema;
- Melhor solução conhecida - Melhor solução conhecida relatada na literatura;
- Limite Inferior - Melhor limitante inferior ou solução primal encontrada;
- Limite Superior - Melhor limitante superior encontrado com a LagClus;
- GAP LB (%) - Gap percentual encontrado para o limitante inferior em relação a solução ótima ou melhor solução conhecida, calculado como $\frac{(SolucaoOtima(melhor) - LimiteInferior)}{SolucaoOtima(melhor)} * 100$;
- GAP UB (%) - Gap percentual encontrado para o limitante superior em relação a solução ótima ou melhor solução conhecida, calculado como $\frac{(LimiteSuperior - SolucaoOtima(melhor))}{SolucaoOtima(melhor)} * 100$;
- Tempo - Tempo em segundos usado pela LagClus;
- Iterações - Número de iterações usadas pela LagClus.

A Tabela 6.1 apresenta os resultados obtidos para as instâncias propostas por Letchford e Amaral (2001) consideradas difíceis para a relaxação lagrangeana. São instâncias que apresentam soluções ótimas com até 50 caixas. Pela tabela, percebe-se que realmente essas instâncias são difíceis, mas a LagClus foi capaz de fechar o *Gap* (Limitante superior - Limitante inferior < 1) para a instância "L7", e nas demais, as soluções ótimas foram obtidas e o limitante dual ficou muito próximo do ótimo, não fechando o *Gap* por uma caixa. O número de *clusters* utilizados foi igual a 2 para todas as instâncias.

TABELA 6.1 – Resultados obtidos com a LagClus para o PCP usando as instâncias de Letchford e Amaral (2001).

						LagClus					
Instância	**L**	**W**	**l**	**w**	**Solução ótima**	**Limite inferior**	**Limite superior**	**GAP LB(%)**	**GAP UB(%)**	**Tempo (s)**	**Iterações**
L1	32	22	5	4	34	34	35,0021	0,00	2,95	50	160
L2	32	27	5	4	42	42	43,0059	0,00	2,39	92	150
L3	40	26	7	4	36	36	37,0016	0,00	2,78	137	160
L4	40	33	7	4	46	46	47,0027	0,00	2,18	357	159
L5	53	26	7	4	48	48	49,0126	0,00	2,11	406	147
L6	37	30	8	3	45	45	46,0402	0,00	2,32	283	159
L7	81	39	9	7	49	49	49,9554	0,00	1,95	575	90
L8	100	64	17	10	36	36	37,0021	0,00	2,78	83	147
L9	100	82	22	8	45	45	46,0137	0,00	2,25	476	150
L10	100	83	22	8	45	45	46,0137	0,00	2,25	474	150

As dez instâncias mostradas na Tabela 6.2 foram obtidas da COVER II proposta por Alvarez-Valdes et al. (2004), e correspondem a problemas em que a solução ótima está entre 50 e 100 caixas. Todas essas instâncias foram obtidas de forma aleatória dentro da COVER II. Como pode ser visto, o limitante dual se aproximou bem da solução ótima, e o limitante primal forneceu resultados bons dado a dificuldade do PCP. O número de *clusters* utilizado foi igual a 5.

TABELA 6.2 – Resultados obtidos com a LagClus para o PCP usando 10 instâncias da COVER II (Alvarez-Valdes et al., 2004).

						LagClus					
Instância	**L**	**W**	**l**	**w**	**Solução ótima**	**Limite inferior**	**Limite superior**	**GAP LB(%)**	**GAP UB(%)**	**Tempo (s)**	**Iterações**
L11	57	53	7	5	85	80	86,3259	5,88	1,56	302	144
L12	84	75	11	6	94	92	95,5313	2,12	1,62	1180	151
L13	151	131	19	11	94	90	94,3784	4,25	0,40	3601	169
L14	61	38	6	5	77	75	77,2133	2,60	0,28	168	165
L15	100	53	9	7	83	80	84,0749	3,75	1,29	389	169
L16	120	80	14	11	61	58	61,9060	4,92	1,49	57	163
L17	51	38	11	3	57	56	58,1986	1,75	2,10	125	168
L18	120	83	17	6	97	94	97,6525	3,09	0,67	1733	165
L19	131	86	16	7	100	96	100,4329	4,00	0,43	2375	144
L20	98	93	17	7	75	73	75,9503	2,67	1,27	591	155

Diante dos resultados mostrados, vale ressaltar os trabalhos de Farago e Morabito (2000) e Morabito e Farago (2002). Em ambos, os autores trabalham com o PCP e utilizam

uma relaxação lagrangeana com otimização do subgradiente para obter bons limitantes superiores. Os autores relaxam todo o conjunto de restrição definido pela Equação 6.16 e adicionam uma única restrição que indica que a soma de todas as variáveis do problema deve ser menor ou igual a $\lfloor (L * W) / (l * w) \rfloor$ (onde $\lfloor z \rfloor$ é o maior inteiro menor ou igual a z). Nesse caso, a equação $\lfloor (L * W) / (l * w) \rfloor$ é um limitante de área, pois a soma de todas as áreas das caixas empacotadas deve ser menor ou igual a área do palete.

Os autores utilizaram dados da COVER I e dados reais obtidos junto às transportadoras brasileiras. Os limitantes obtidos foram bons, no entanto, não foi possível fazer comparações pois os problemas utilizados foram diferentes dos apresentados nesta proposta.

Seguindo a mesma linha de Farago e Morabito (2000) e Morabito e Farago (2002), Oliveira (2005) trabalhou com o mesmo tipo de relaxação lagrangeana e com a mesma restrição de área, além de um *Branch-and-Bound*. Ela utilizou em seu trabalho, entre outras instâncias, as de Letchford e Amaral (2001) mostradas na Tabela 6.1. Apenas com a relaxação lagrangeana, em nenhuma dessas instâncias Oliveira (2005) conseguiu encontrar um limite superior que pudesse comprovar uma solução ótima, diferentemente da LagClus. Esse fato comprova mais uma vez que a LagClus é uma relaxação "forte", podendo ser muito útil em problemas que podem ser modelados através de grafo de conflitos. Por outro lado, os limitantes obtidos por Oliveira (2005) não puderam ser comparados aos obtidos pela LagClus, pois a autora implementa um algoritmo de otimização do subgradiente que na atualização do passo, considera a parte inteira do melhor limitante superior.

TABELA 6.3 – Resultados obtidos com a LagClus para o PCP usando 10 instâncias da COVER III (Alvarez-Valdes et al., 2004).

Instância	**L**	**W**	**l**	**w**	**Melhor solução conhecida**	**LagClus**					
						Limite inferior	**Limite superior**	**GAP LB(%)**	**GAP UB(%)**	**Tempo (s)**	**Iterações**
L21	99	88	12	5	144	136	145,6033	5,56	1,11	1136	144
L22	99	75	13	5	113	110	114,3892	2,65	1,23	328	144
L23	97	95	9	7	145	140	146,6249	3,45	1,12	685	144
L24	98	98	10	7	136	135	137,5787	0,74	1,16	536	144
L25	98	88	10	7	122	116	123,5109	4,92	1,24	297	144
L26	97	96	11	6	140	136	141,6190	2,86	1,16	873	144
L27	96	87	8	7	148	144	149,5781	2,70	1,07	513	144
L28	99	70	15	4	114	110	115,8749	3,51	1,64	334	144
L29	91	70	12	5	105	103	106,3093	1,90	1,25	199	144
L30	93	84	11	6	117	112	118,6137	4,27	1,38	406	144

A Tabela 6.3 apresenta os resultados obtidos com a LagClus para 10 instâncias da COVER III. Nesse caso, as soluções ótimas estão entre 100 e 150 caixas, e o número

de *clusters* utilizado foi igual a 15 para todas as instâncias.

Essas instâncias foram obtidas de forma semelhante ao que foi feito nas instâncias da Tabela 6.2, porém, elas foram selecionadas aleatoriamente a partir do conjunto das instâncias que não apresentam solução ótima conhecida.

6.5 Resultados obtidos com a geração de colunas

Mais uma vez a LagClus apresentou resultados interessantes, fornecendo limitantes superiores muito próximos da melhor solução, entretanto, não foi garantida a optimalidade de nenhuma solução conhecida.

A decomposição apresentada na Seção 6.1 foi aplicada no PCP para as instâncias "L1"-"L10" com 2 *clusters*. Os resultados estão na Tabela 6.4. O processo de geração de colunas foi feito até que nenhuma coluna de custo reduzido positivo fosse inserida no PMR.

As colunas presentes na Tabela 6.4 referem-se a:

- Instância - Instância considerada;
- Número inicial de colunas - Número inicial de colunas obtidas com a heurística RSF;
- Resultado inicial (PMR) - Valor da função objetivo para o PMR, considerando somente o conjunto inicial de colunas;
- Número de final de colunas - Número final de colunas obtido quando o processo de geração foi encerrado;
- Resultado final (PMR) - Valor da função objetivo para o PMR com o número final de colunas;
- Tempo1(s) - Tempo em segundos utilizado pelo processo de geração de colunas;
- Solução - Representa o valor da função objetivo para o PMR final convertido em um problema inteiro;
- Tempo2(s) - Tempo em segundos para resolver o PMR final convertido em um problema inteiro.

Para gerar o conjunto inicial de colunas, foi utilizada uma adaptação da heurística RSF (*Recursive-Smallest-First*), proposta por Yamamoto (2003), que foi baseada na heurística RLF (*Recursive-Largest-First*) (ver Klotz (2002)), proposta para o problema de coloração de grafos.

Dado uma lista de vértices ativos, a heurística RSF inicia escolhendo um vértice x de grau mínimo, em seguida, ela torna o vértice escolhido e seus vértices adjacentes inativos. A partir da lista de vértices ativos, é calculado novamente o grau de cada vértice, e em seguida um novo vértice de grau mínimo é selecionado, e o ciclo é então repetido. O algoritmo termina quando não houver mais vértices ativos. Os vértices escolhidos formam um conjunto independente de vértices.

A RSF foi usada da seguinte maneira para gerar o conjunto inicial de colunas do PCP. Ao invés de escolher o vértice de grau mínimo, escolhe-se aleatoriamente um vértice x. Em seguida, o vértice escolhido e seus vértices adjacentes são colocados inativos, e daí por diante, o processo é idêntico ao original. Quando a heurística termina, os vértices selecionados formam uma solução para o PCP. Todo esse processo é repetido até que se obtenha o número desejado de soluções. Sendo ND esse número, o número de colunas adicionadas no PMR é igual a $ND * \overline{P}$, pois cada *cluster* gera uma coluna.

TABELA 6.4 – Resultados obtidos com a geração de colunas.

Instância	**Geração de colunas**					**Resolvendo o PMR final de forma inteira (PLI)**	
	Número inicial de colunas	**Resultado inicial (PMR)**	**Número final de colunas**	**Resultado final (PMR)**	**Tempo1(s)**	**Solução**	**Tempo2(s)**
L1	500	34,00	626	35,00	17	34	0,00
L2	500	41,00	674	43,00	45	42	0,20
L3	500	35,16	697	37,00	62	36	0,00
L4	500	45,00	680	47,00	183	46	2,00
L5	500	47,20	691	49,00	286	48	1,00
L6	500	44,00	829	46,00	194	45	0,00
L7	500	49,00	815	49,85	2166	49	1,00
L8	500	36,00	645	37,00	27	36	0,00
L9	500	44,00	813	46,00	331	45	4,01
L10	500	44,00	813	46,00	326	45	5,10

Pela Tabela 6.4 verifica-se que todas as soluções inteiras obtidas após o processo de geração de colunas, foram iguais as soluções ótimas desses problemas, em um tempo computacional baixo, menor que 5,10 segundos.

A decomposição Dantzig-Wolfe mostrada utiliza como geradores de colunas, os *clusters* formados a partir do particionamento. Assim, a LagClus pode ser obtida de forma

indireta, através desses subproblemas, considerando as informações duais do PMR.

6.6 Considerações finais

Este capítulo apresentou a decomposição Dantzig-Wolfe para o problema de máximo conjunto independente de vértices, usando a divisão em *clusters*. Na sequência, foi apresentado o problema do carregamento de paletes que foi resolvido pela LagClus, e pelo método de geração de colunas.

A LagClus foi capaz de resolver uma instância considerada difícil para uma relaxação lagrangeana, e seus *Gaps* foram pequenos, mostrando ser uma relaxação "forte". Por outro lado, a geração de colunas também apresentou resultados interessantes, no entanto, o tempo para gerar todas as colunas foi, em alguns casos, alto como por exemplo para a instânca "L7". Isso demonstra a necessidade de um estudo mais detalhado sobre o processo de geração de colunas, permitindo eliminar colunas improdutivas e com isso, reduzir este tempo.

No entanto, pode-se dizer que a primeira parte do algoritmo B&P proposto está fornecendo bons resultados. É de se esperar que os resultados obtidos com o B&P sejam comparáveis com os melhores resultados encontrados na literatura para o PCP.

CAPÍTULO 7

ATIVIDADES PREVISTAS, CRONOGRAMA E CONCLUSÕES

Neste capítulo são apresentados as próximas atividades para o *Branch-and-Price* e para a LagClus, considerando os problemas da rotulação cartográfica de pontos, do carregamento de paletes e da programação diária de fotos de satélite.

7.1 LagClus

A LagClus apresentou bons resultados para o PRCP e para o PCP mostrados nesta proposta. Pretende-se assim, aplicá-la na formulação reduzida obtida para o problema da programação de fotos de satélite. Será feito um estudo para tentar não relaxar as cliques presentes na formulação relaxada, assim como foi feito para o modelagem baseada em pontos do PRCP.

Também será feito um estudo mais detalhado sobre a fase do particiomento de grafos, pois, como já foi dito, dependendo no número de arestas entre *clusters*, a relaxação obtida pode ser mais ou menos "forte".

Outro ponto importante relacionado ao particionamento do grafo, diz respeito ao número ótimo de *clusters*. Não foi verificado ainda alguma relação entre este número e as propriedades do grafo como número de vértices, arestas e densidade. Pretende-se fazer um levantamento para verificar se existe algum estudo e se é possível estimar esse número ótimo de *clusters*.

7.2 *Branch-and-Price*

O *Branch-and-Price* mostrado encontra-se em sua fase inicial de desenvolvimento, a de geração de colunas. Para torná-lo mais eficiente, pretende-se estudar algumas técnicas de eliminação de colunas improdutivas e obter, ao final do processo de geração de colunas, um PMR menor porém de melhor qualidade. Está em fase de estudo a remoção de colunas usando como critério a média dos custos reduzidos. Porém, como cada *cluster* produz uma coluna com seu respectivo custo reduzido, o que está sendo estudado é a remoção de colunas feita em cada *cluster*, considerando a sua respectiva média. Isso é particular desta decomposição, pois normalmente utiliza-se a eliminação de colunas considerando a média total dos custos reduzidos.

Além disso, questões associadas ao término do processo de geração de colunas serão estudados, inclusive a análise do uso do limitante dual fornecido pela LagClus para

realizar tal tarefa. Para o PCP existem limitantes teóricos que podem ser obtidos facilmente tais como o limitante de área, e que também podem ser utilizados para parar o processo de geração de colunas.

Na segunda fase do método, pretende-se estudar as estratégias existentes de separação e descida na árvore de busca, objetivando realizar a maior parte das podas na parte superior da árvore. Também serão estudadas as técnicas existentes para a geração de colunas em cada nó da árvore. Espera-se obter um método eficiente capaz de resolver problemas de larga escala relacionados ao PRCP, PCP e ao PPDF.

7.3 Aplicações: atividades específicas

- Problema da Rotulação Cartográfica de Pontos. Para utilizar o *Branch-and-Price* em desenvolvimento neste problema, será necessário antes estudar como será o seu PMR devido às restrições de adjacências presentes explicitamente na formulação. Em seguida, serão utilizadas as instâncias de Yamamoto e Lorena (2005) para avaliar o método;

- Problema do Carregamento de Paletes. Aplicar a LagClus em todas as instâncias da COVER I, II e III propostas por Alvarez-Valdes et al. (2004), e estudar uma heurística de verificação e melhoria que seja melhor do que a que foi desenvolvida inicialmente. Em seguida, aplicar o *Branch-and-Price* proposto nesse problema sobre as mesmas instâncias usadas na LagClus. Isso permitirá uma análise comparativa dos resultados;

- Programação de Fotos de Satélite. Este problema apresenta características de um PMCIV mas no entanto, não pode ser classificado como tal. Entretanto, a LagClus será aplicada seguindo a linha da aplicação feita na modelagem baseada em pontos para o PRCP, considerando a formulação reduzida apresentada neste trabalho. Espera-se obter resultados tão bons quanto os obtidos por Vasquez e Hao (2003).

7.4 Cronograma

O cronograma de atividades mostrado abaixo envolve atividades específicas, relativas às aplicações, e gerais como revisão bibliográfica e redação do texto.

A Tabela 7.1 apresenta o cronograma para os períodos letivos 2 e 3 de 2005. Na Tabela 7.2, é apresentado o cronograma dos períodos letivos 1, 2 e 3 de 2006 e por último, na Tabela 7.3, é apresentado o cronograma para o primeiro período de 2007, etapa final que constitui a defesa da tese.

Basicamente, o restante de 2005 será dedicado ao estudo do *Branch-and-Price*, e em 2006, iniciando 2007, serão feitas as aplicações tanto da LagClus quanto do *Branch-and-Price* nos problemas apresentados nesta proposta de tese.

TABELA 7.1 – Cronograma para o segundo e terceiro período letivo de 2005.

2005						
	JUL	AGO	SET	OUT	NOV	DEZ
Análise de parada (B&P)	X	X	X			
Eliminação de colunas (B&P)	X	X	X	X		
Estratégia de descida (B&P)			X	X	X	
Estratégia de podas (B&P)				X	X	X
Geração de colunas nos nós (B&P)					X	X
Revisão bibliográfica	X	X		X	X	X
Redação do texto				X	X	X
Submeter artigos para publicação/congresso	X					X

TABELA 7.2 – Cronograma para o ano letivo de 2006.

2006												
	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT	NOV	DEZ
PCP												
Testes (LagClus / B&P)	X	X	X									
Avaliação	X	X	X	X								
PRCP												
Abordagem (B&P)			X	X	X	X	X					
Testes (B&P)					X	X	X					
Avaliação						X	X	X	X			
PPDF												
Abordagem (LagClus)								X	X	X		
Testes (LagClus)									X	X	X	
Avaliação										X	X	X
Revisão bibliográfica	X		X	X	X			X	X	X		
Redação do texto	X	X	X	X	X	X		X	X	X	X	
Submeter artigos para publicação/congresso			X			X			X			X

TABELA 7.3 – Cronograma para o primeiro período letivo de 2007.

2007						
	JAN	FEV	MAR	ABR	MAI	JUN
Redação do texto	X	X	X			
Texto final				X		
Submeter artigos para publicação/congresso			X			X
Entrega para banca					X	
Defesa						X

7.5 Conclusões

Esta proposta apresentou os primeiros resultados obtidos com a relaxação lagrangeana com *clusters* e com a geração de colunas. Inicialmente, fundamentos teóricos sobre a relaxação lagrangeana e particionamento de grafos foram mostrados, pois estes formam a base da LagClus.

A LagClus permite explorar, principalmente, problemas que podem ser representados e formulados através de grafos de conflitos. A idéia de particionar o problema, fornece uma relaxação mais forte, dado que os subproblemas gerados têm características semelhantes ao problema original. Outra questão importante embutida nessa relaxação, é a idéia de obter subproblemas separáveis facilmente resolvidos.

A LagClus foi testada nos problemas de rotulação cartográfica e carregamento de paletes. Os resultados obtidos foram satisfatórios, mostrando ser esta uma relaxação interessante para esses problemas.

O problema da programação diária de fotos também foi estudado, e uma formulação reduzida foi apresentada. Este problema possui características de um problema de máximo conjunto independente de vértices, mas não pode ser definido como tal. Assim, com a aplicação da LagClus neste problema, poderá ser possível avaliar seu comportamento em problemas não totalmente definidos como pertencentes a classe de máximo conjunto independente.

Por último, foi apresentada a decomposição Dantzig-Wolfe para o problema do carregamento de paletes, usando a divisão em *clusters*. Os resultados encontrados justificam esta proposta de tese de desenvolver um *Branch-and-Price* para esse e para o problema da rotulação.

Esta proposta já apresenta algumas constribuições. No problema da rotulação cartográfica de pontos, a abordagem de minimização do número de conflitos, as duas formulações, e todas as relaxações, inclusive a LagClus, ainda não tinham sido exploradas na literatura. No problema do carregamento de paletes, tanto a aplicação da LagClus, quanto a geração de colunas usando a decomposição Dantzig-Wolfe mostrada, também não tinham sido exploradas, assim como a formulação reduzida obtida para o problema da programação diária de fotos de satélite.

Com isso, espera-se obter outros resultados interessantes com os estudos e análises apresentadas nas seções anteriores deste capítulo.

REFERÊNCIAS BIBLIOGRÁFICAS

Aarts, E.; Korst, J. **Simulated Annealing and Boltzmann Machines**. Chichester, UK: J. Wiley & Sons, 1989. 28

Alon, N.; Babai, L.; Itai, A. A fast and simple randomized parallel algorithm for the maximal independent set problem. **Journal of Algorithms**, v. 7, n. 4, p. 567–583, December 1986. 28

Alvarez-Valdes, R.; Parreño, F.; Tamarit, J. M. A branch-and-cut algorithm for the pallet loading problem. **Computers and Operations Reserach**, 2004. In press. 10, 15, 73, 80, 84, 86

Atamtürk, A.; Nemhauser, G. L.; Savelsbergh, M. W. P. Conflict graphs in solving integer programming problems. **European Journal of Operational Research**, v. 121, p. 40–55, 2000. 15

Balas, E.; Xue, J. Minimum weighted coloring of triangulated graphs, with application to maximum weight vertex packing and clique finding in arbitrary graphs. **SIAM Journal on Computing**, v. 20, n. 2, p. 209–221, April 1991. 27

——. Weighted and unweighted maximum clique algorithms with upper bounds from fractional coloring. **Algorithmica**, v. 15, n. 5, p. 397–412, May 1996. 27

Balas, E.; Yu, C. S. Finding a maximum clique in an arbitrary graph. **SIAM Journal Computing**, v. 15, p. 1054–1068, 1986. 27

Barnes, E. R. A branch-and-bound procedure for the largest clique in a graph. In: Pardalos, P. M. (ed.) **Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems**. Boston: Kluwer Academic Publishers, 2000. 27

Barnhart, C.; Jhonson, E. L.; Nemhauser, G. L.; Savelsberg, M. W. P.; Vance, P. H. Branch-and-Price: column generation for solving huge integer programs. **Operations Research**, v. 46, p. 316–329, 1998. 69

Bazaraa, M. S.; Jarvis, J. J.; Sherali, H. D. **Linear programming and network flows**. New York: John Wiley & Sons, 1990. 2nd edition. 27

Beasley, J. An exact two-dimensional non guillotine cutting tree search procedure. **Operations Research**, v. 33, p. 49–64, 1985. 73, 74, 78

——. A lagrangean heuristic for set-covering problems. **Naval Research Logistics**, v. 37, p. 151–164, 1990. 20

Bensana, E.; Lemaitre, M.; Verfaillie, G. Benchmark problems: earth observation satellite management. **Contraints**, v. 4, n. 3, p. 293–299, 1999. 57, 64

Bhattacharya, R.; Roy, R.; Bhattacharya, S. An exact depth-first algorithm for the pallet loading problem. **European Journal of Operational Research**, v. 110, p. 610–625, 1998. 73

Björkund, P.; Värbrand, P.; Yuan, D. Optimized planning of frequency hopping in cellular networks. **Computers and Operations Reserach**, v. 32, p. 169–186, 2005. 15

Bomze, I. M.; Budinich, M.; Pardalos, P. M.; Pelilo, M. The maximum clique problem. In: Du, D.; Pardalos, P. M. (eds.) **Handbook of Combinatorial Optimization**. Boston: Kluwer Academic Publishers, 1999. 27

Bourjolly, J. M.; Laporte, G.; Mercure, H. A combinatorial column generation algorithm for the maximum stable set problem. **Operations Research Letters**, v. 20, p. 21–29, 1997. 27

Brigham, R. D.; Dutton, R. D. A new graph coloring algorithm. **The Computer Journal**, v. 24, p. 85–86, 1981. 27

Bron, C.; Kerbosch, J. Finding all cliques of an undirected graph. **Communications of the ACM**, v. 16, n. 9, p. 575–575, September 1973. 27

Bui, T. N.; Eppley, P. H. A hybrid genetic algorithm for the maximum clique problem. In: Proceedings 6th International Conference on Genetic Algorithms. **Anais...** 1995. p. 478–484. 28

Bui, T. N.; Moon, B. R. Genetic algorithm and graph partitioning. **IEEE Transactions on Computers**, v. 45, n. 7, p. 841–855, 1996. 23

Carraghan, R.; Pardalos, P. M. An exact algorithm for the maxmimum clique problem. **Operations Research Letters**, v. 9, p. 375–382, 1990. 27

Christensen, J.; Marks, J.; Shieber, S. **Placing text labels on maps and diagrams**. London: Academic Press, 1993. 39

——. An empirical study of algorithms for point-features label placement. **ACM Transactions on Graphics**, v. 14, n. 3, p. 203–232, 1995. 7, 37, 38, 40, 54

Churh, R. L.; Murray, A. T.; Weintraub, A. Locational issues in forest management. **Location Science**, v. 6, p. 137–153, 1998. 15

Dantzig, G. B.; Wolfe, P. Decomposition principle for linear programs. **Operations Research**, v. 8, p. 101–111, 1960. 69

Desaulniers, G.; Desrosiers, J.; Solomon, M. M. Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems. In: Hansen, P.; Ribeiro, C. (eds.) **Essays and Surveys in Metaheuristics**. Kluwer Academic Publishers, 2001. 69

Desrochers, M.; Desrosiers, J.; Solomon, M. A new optimization algorithm for the vehicle routing problem with time windows. **Operations Research**, v. 40, p. 342–354, 1992. 69

Dowsland, K. A. An exact algorithm for the pallet loading problem. **European Journal of Operational Research**, v. 31, p. 78–84, 1987. 15, 73

Dyckhoff, H. A typology of cutting and packing problems. **European Journal of Operational Research**, v. 31, p. 78–84, 1990. 72

Farago, R.; Morabito, R. Um método heurístico baseado em relaxação lagrangiana para o problema de carregamento de paletes do produtor. **Pesquisa Operacional**, v. 20, n. 2, p. 197–212, 2000. 80, 81

Feo, T. A.; Resende, M. G. C.; Smith, S. H. A greedy randomized adaptive search procedure for maximum independent set. **Operations Research**, v. 42, p. 860–878, 1994. 28

Fjällström, P. O. **Algorithms for graph partitioning: a survey**. Computer and Information Science: Linköping University, 1998. Vol. 3, Nr. 10. Disponível em: <http://www.ep.liu.se/ea/cis/1998/010>. Acesso em: 22 jan. 2004. 23, 25

Garey, M.; Johnson, D.; Stockmeyer, L. Some simplified NP-complete graph problems. **Theoretical Computer Science**, v. 1, p. 237–267, 1976. 16, 22

Gendreau, M.; Salvail, L.; Soriano, P. Solving the maximum clique problem using a tabu search approah. **Annals of Operations Research**, v. 41, p. 385–403, 1999. 28

Gerrard, R.; Church, R. L. Closest assignament constraints and location models: properties and structures. **Location Science**, v. 4, p. 251–270, 1996. 15

Gilbert, J. R.; Miller, G. L.; Teng, S.-H. Geometric mesh partitioning: implementations and experiments. In: Proc. International Parallel Processing Symposium, 1995. **Anais...** 1995. p. 418–427. 25

Gilbert, J. R.; Zmijewski, E. A parallel graph partitioning algorithm for a message-passing multiprocessor. **International Journal of Parallel Programming**, v. 16, n. 1, p. 427–449, 1987. 25

Glover, F. Surrogate constraints. **Operations Research**, v. 16, p. 741–749, 1968. 43

Godbeer, G. H.; Lipscomb, J.; Luby, M. **On the computational complexity of finding stable state vectors in connectionist models (Hopfield nets)**. Department of Computer Science: University Toronto, 1988. Technical Report 208/88. 28

Goycoolea, M.; Murray, A. T.; Barahona, F.; Epstein, R.; Weintraub, A. Harvest scheduling subject to maximum area restrictions: exploring exact approaches. **Operations Research**, v. 1, 2005. To appear. 15, 16

Harary, F. **Graph theory**. New York: Addison-Wesley, 1972. 15, 51

Held, M.; Karp, R. M. The traveling salesman problem and minimum spanning trees: part II. **Mathematical Programming**, v. 1, p. 6–25, 1971. 20, 43

Hendrickson, B.; Leland, R. **The CHACO user's guide**. Sandia National Laboratories: Albuquereque, NM, 1995. SAND94–2692. Disponível em: <http://www.cs.purdue.edu/research/cse/related/decompose/chaco.html>. Acesso em: 01 abr. 2005. 25

Herbert, A.; Dowsland, W. A family of genetic algorithms for the pallet loading problem. **Annals of Operations Research**, v. 63, p. 415–436, 1996. 73

Hertz, A. A fast algorithm for coloring Meyniel graphs. **Journal of Combinatorial Theory B**, v. 50, n. 2, p. 231–240, 1990. 27

Hicks, I. V.; Warren, J. S.; Warrier, D.; Wilhenlm, W. E. **A Branch-and-Price approach for the maximum weight independent set problem**. Texas A & M University: Department of Industrial Engineering, 2004. Disponível em: <http://ie.tamu.edu/People/faculty/Hicks/>. Acesso em: 22 jan. 2004. 27, 69, 71

Hirsh, S. A. An algorithm for automatic name placement around point data. **American Cartographer**, v. 9, n. 1, p. 5–17, 1982. 54

ILOG CPLEX 7.5 Reference Manual. 7.5v. 610 p. ©Copyright by ILOG, France, 2001. 9, 42, 52

Johnson, D. S.; Aragon, C. R.; McGeoch, L. A.; Schevon, C. Optimization by simulated annealing: an experimental evaluation; part I, graph partitioning. **Operations Research**, v. 37, n. 6, p. 865–892, 1989. 23

Karisch, S. E.; Rendl, F.; Clausen, J. **Solving graph bi-section problems with semidefinite programming**. Departament of Computer Science: University of Copenhagen, 1997. DIKU-TR-97/9. Disponível em: <http://ie.tamu.edu/People/faculty/Hicks/>. Acesso em: 22 jan. 2004. 23

Karypis, G. **Multilevel hypergraph partitiong**. Departament of Computer Science: University of Minnesota, 2002. TR 02–25. Disponível em: <http://www-users.cs.umn.edu/~karypis/publications/partitioning.html>. Acesso em: 30 mar. 2005. 7, 24

Karypis, G.; Kumar, V. A coarse-grain parallel formulation of multilevel k-way partitioning algorithm. In: Proc. Eighth SIAM Conference on Parallel Processing for Scientific Computing, 1997. **Anais...** 1997. 25

——. A fast and high quality multilevel scheme for patitioning irregualar graphs. **SIAM Journal on Scientific Computing**, v. 20, p. 359–392, 1998a. 17, 24, 25

——. **Multilevel algorithms for multi-constraint graph partitiong**. Army HPC Research Center, Dep. of Computer Science: University of Minnesota, 1998b. Technical Report 98-019. 25

——. Multilevel k-way partitioning scheme for irregular graphs. **Journal of Parallel and Distributed Computing**, v. 48, n. 1, p. 96–129, 1998c. 24, 25, 49

——. Multilevel k-way hypergraph partitioning. In: Proceedings of the Design and Automation Conference, 1999. **Anais...** 1999. 22

Kernighan, B. W.; Lin, S. An efficient heuristic procedure for partitioning graphs. **The Bell System Technical J.**, v. 49, p. 291–307, 1970. 23, 25

Klotz, W. Graph Coloring Algorithms. **Mathematik-Bericht**, v. 5, p. 1–9, 2002. TU Clausthal. 15, 82

Kopf, R.; Ruhe, G. A computational study of the weighted independent set problem for general graphs. **Found. Control Engin.**, v. 12, n. 4, p. 167–180, 1987. 28

van Krieken, M. G. C.; Fleuren, H. A.; Peeters, M. J. P. **A lagrangean relaxation based algorithm for solving set partitioning problems**. CentER Discussion:

Tilburg University, 2004. No–2004–44. Disponível em: <http://ssrn.com/abstract=557848>. Acesso em: 12 nov. 2004. 20

Leighton, F. T. A graph coloring algorithm for large scheduling problems. **Journal of Research of the National Bureau of Standards**, v. 84, p. 489–503, 1979. 27

Letchford, A. N.; Amaral, A. Analysis of upper bounds for the pallet loading problem. **European Journal of Operational Research**, v. 3, n. 132, p. 582–593, 2001. 9, 16, 28, 80, 81

Lins, L.; Lins, S.; Morabito, R. A L-approach for packing (l-w)rectangles into rectangular and L-shaped pieces. **Journal of the Operational Research Society**, v. 54, p. 777–789, 2003. 73

Lorena, L. A. N.; Furtado, J. C. Constructive Genetic Algorithm for clustering problems. **Evolutionary Computation**, v. 3, n. 9, p. 309–327, 2001. 37, 40

Lorena, L. A. N.; Senne, E. L. F. Improving traditional subgradient scheme for Lagrangean relaxation: an application to location problems. **International Journal of Mathematical Algorithms**, v. 1, p. 133–151, 1999. 20

Lorena, L. A. N.; Senne, L. F. A column generation approach to capacited p-median problems. **Computers and Operations Research**, v. 31, p. 863–876, 2003. 69

Mauri, G. R.; Lorena, L. A. N. Driver scheduling generation using a population training algorithm. In: SBRN 04 - Brazilian Symposium in Neural Networks, 2004, São Luiz. **Anais...** Maranhão: SBRN, 2004. 1 CD-ROM. Disponível em: <http://www.lac.inpe.br/~lorena/mauri/driver-pta.pdf>. Acesso em: 10 fev. 2005. 69

Mehrotra, A.; Trick, M. A. A column generation approach for graph coloring. **INFORMS Journal on Computing**, v. 8, p. 344–354, 1996. 27

Miller, G. L.; Teng, S.-H.; Thurston, W.; Vavasis, S. A. Graph theory and sparse matriz computation. In: George, A.; Gilbert, J. R.; Liu, L. W. H. (eds.) **The IMA Volumes in Mathematics and its Applications**. Springer Verlag, 1993. p. 57–84. 23

Mobasher, B.; Jain, N.; Han, E.; Srivastava, J. **Web mining: pattern discovery from world wide web transactions**. Department of Computer Science: University of Minnesota, 1996. TR–96–050. 22

Moon, I. D.; Chaudhry, S. An analysis of network location problems with distance constraints. **Management Science**, , n. 30, p. 290–307, 1984. 39

Morabito, R.; Farago, R. A tight lagrangean relaxation bound for the manufacturer's pallet loading problem. **Studia Informatica Universalis**, v. 2, n. 1, p. 57–76, 2002. 80, 81

Murray, A. T.; Church, R. L. Constructing and selecting adjacency constraints. **INFOR**, p. 232–248, 1996. 16

——. Facets for node packing. **European Journal of Operational Research**, v. 101, p. 598–608, 1997a. 7, 8, 16, 39, 41, 60, 61, 63, 64

——. Solving the anti-covering location problem using lagrangean relaxation. **Computers and Operations Research**, v. 24, n. 2, p. 127–140, 1997b. 15, 16, 20

Narciso, M. G. **A Relaxação lagrangeana/surrogate e algumas aplicacoes em otimização combinatória**. 1998. Tese (Doutorado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos. 1998. 21

Narciso, M. G.; Lorena, L. A. N. Lagrangean/Surrogate relaxation for generalized assignment problems. **European Journal of Operational Research**, v. 114, p. 165–177, 1999. 16, 43

Oliveira, L. K. **Métodos exatos baseados em relaxações lagrangiana e surrogate para o problema de carregamento de paletes do produtor**. 2005. Tese (Doutorado em Engenharia de Produção) – Universidade de São Paulo, São Carlos. 2005. 75, 81

Papadimitriou, C. H.; Steiglitz, K. **Combinatorial optimization**. New York: Dover, 1998. 19

Parker, R. G.; Rardin, R. L. **Discrete optimization**. New York: Academic Press, 1988. 19, 21, 22

Pureza, V.; Morabito, R. Some experiments with a simple tabu search algorithm for the manufacturer's pallet loading problem. **Computers and Operations Research**, 2005. To appear. 15, 73

Ribeiro, G. M.; Lorena, L. A. N. Heuristics for cartographic label placement problems. **Computers and Geoscience**, 2004a. Submitted. 37

——. Lagrangean relaxation bounds for point-feature cartographic label placement problem. **Pesquisa Operacional**, 2004b. Submitted. 41

——. Lagrangean relaxation with clusters for point-feature cartographic label placement problems. **Computers and Operations Research**, 2004c. Submitted. 15, 37, 50

——. Modelagem matemática e relaxações Lagrangeana e Lagrangeana/Surrogate para o problema de rotulação cartográfica de pontos. In: Simpósio Brasileiro de Pesquisa Operacional (SBPO), 36., 2004, São João del Rei. **Anais...** Rio de Janeiro: SOBRAPO, 2004d. 1 CD-ROM. Disponível em: <http://www.lac.inpe.br/~lorena/glaydston/Glaydston-lorena-Sbpo.PDF>. Acesso em: 20 jan. 2005. 40

——. Relaxação Lagrangeana com formação de clusters aplicada a rotulação cartográfica de pontos. In: VIII Oficina Nacional sobre problemas de corte e empacotamento e correlatos, 2004, São José dos Campos. **Proceedings...** São Paulo: INPE, 2004e. Disponível em: <http://www.lac.inpe.br/VIIIPCE/index.htm>. Acesso em: 20 jan. 2005. 40, 50

Rolland, E.; Pirkul, H.; Glover, F. Tabu search for graph partitioning. **Annals of Operations Research**, v. 63, p. 209–232, 1996. 23

Scheithauer, G.; Terno, J. The G4-heuristic for the pallet loading problem. **Journal of the Operational Research Society**, v. 47, p. 511–522, 1996. 73

Schloegel, K.; Karypis, G.; Kumar, V. **Multilevel diffusion schemes for repartitioning of adaptive meshes**. Department of Computer Science: University of Minnesota, 1997. TR–97–013. 25

Schreyer, M.; Raidl, G. R. Letting Ants Labeling Point Features. In: 2002 IEEE Congress on Evolutionary Computation at the IEEE World Congress on Computational Intelligence. **Anais...** 2004. p. 1564–1569. 40

Senne, E. L. F.; Lorena, L. A. N. Lagrangean/Surrogate heuristics for p-median problems. In: Laguna, M.; Gonzalez-Velarde, J. L. (eds.) **Computing Tools for Modeling, Optimization and Simulation:Interfaces in Computer Science and Operations Research**. Netherlands: Kluwer Academic Publisher, 2000. p. 115–130. 44, 47

Senne, L. F.; Lorena, L. A. N. Stabilizing column generation using lagrangean/surrogate relaxation. In: EURO 2001 - The European Operational Research Conference, 2001, Erasmus University. **Proceedings...** Rotterdam, 2001. 1 CD-ROM. Disponível em:

<http://www.lac.inpe.br/~lorena/ejor/EURO2001.pdf>. Acesso em: 10 fev. 2005. 69

Senne, L. F.; Lorena, L. A. N.; Pereira, M. A. A branch-and-price approach to p-median location problems. **Computers and Operations Research**, v. 32, p. 1655–1664, 2005. 69

Shekhar, S.; Liu, D. R. Partitioning similarity graphs: a framework for declustering problems. **Information Systems Journal**, v. 21, n. 4, 1996. 22

Simon, H. D. Partitioning of unstructured problems for parallel processing. **Computing Systems in Engineering**, v. 2, n. 2–3, p. 135–148, 1991. 24

Simon, H. D.; Farhat, C. **TOP/DOMDEC, a software for mesh partitioning and parallel processing**. NASA, 1993. RNR–93–011. 25

Soriano, P.; Gendreau, M. Diversification strategies in tabu search algorithms for the maximum clique problem. **Annals of Operations Research**, v. 63, p. 189–207, 1996. 28

Östergard, P. R. J. A fast algorithm for the maxmimum clique problem. **Discrete Applied Mathematics**, v. 120, p. 197–207, 2002. 27

Strijk, T.; Verweij, B.; Aardal, K. **Algorithms for maximum independent set applied to map labeling**. Universiteit Utrecht: Institute of Information and Computing Sciences, 2000. Disponível em: <ftp://ftp.cs.uu.nl/pub/ruu/cstechreps/cs-2000/2000-22.ps.gz>. Acesso em: 10 nov. 2003. 39

Vasquez, M.; Hao, J. K. A logic-constrained knapsack formulation and a tabu algorithm for daily photograph scheduling of an earth observation satellite. **Journal of Computational Optimization and Applications**, v. 20, p. 137–157, 2001. 57, 59, 65, 66, 67

——. Upper bounds for the SPOT 5 daily photograph scheduling problem. **Journal of Combinatorial Optimization**, v. 7, p. 87–103, 2003. 57, 67, 86

Verfaillie, G.; Lemaitre, M.; Schiex, T. Russian doll search for solving constraint optimization problems. In: Proc. 13th National Conference on Artificial Intelligence, 1996. **Anais...** Portland, USA, 1996. p. 182–187. 57

Verner, O. V.; Wainwright, R. L.; Schoenefeld, D. A. Placing text labels on maps and diagrams using genetic algorithms with masking. **INFORMS Journal on Computing**, v. 9, p. 266–275, 1997. 16, 40, 54

Walshaw, C. Multilevel refinement for combinatorial optimisation problems. **Annals of Operations Research**, v. 131, p. 325–372, 2004. 25

Walshaw, C.; Cross, M.; Everett, M. G. Parallel dynamic graph partitioning for adaptive unstructured meshes. **Journal of Parallel and Distributed Computing**, v. 47, n. 2, p. 102–108, 1997. Disponível em: <http://www.gre.ac.uk/~c.walshaw/jostle/>. 25

Wolsey, L. A. **Integer programming**. New York: Jhon Wiley & Sons, 1998. 7, 20, 21, 69

Wood, D. R. An algorithm for finding a maximum clique in a graph. **Operations Research Letters**, v. 21, p. 211–217, 1997. 27

Yamamoto, M. **Novos algoritmos para o problema de rotulacao cartografica de pontos**. 2003. Tese (Doutorado em Computação e Matemática Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos. 2003. 82

Yamamoto, M.; Camara, G.; Lorena, L. A. N. Tabu search heuristic for point-feature cartographic label placement. In:_____. **GeoInformatica. An International Journal on Advances of Computer Science for Geographic Information Systems**. Netherlands: Kluwer Academic Publisher, 2002. p. 77–90. 15, 40, 42, 46, 54

Yamamoto, M.; Lorena, L. A. N. A constructive genetic approach to point-feature cartographic label placement. In: Ibaraki, T.; Nonobe, K.; Yagiura, M. (eds.) **Metaheuristics: Progress as Real Problem Solvers**. Netherlands: Kluwer Academic Publishers, 2005. p. 285–300. 15, 16, 37, 40, 41, 42, 46, 51, 54, 86

Young-Gun, G.; Maing-Kyu, K. A simple and effective procedure for the manufacturer's pallet loading problem. **Journal of the Operational Research Society**, v. 49, p. 819–828, 1998. 73

——. A fast algorithm for two-dimensional pallet loading problems of large size. **European Journal of Operational Research**, v. 134, p. 193–202, 2001. 73

Zoraster, S. Integer programming applied to the map label placement problem. **Cartographica**, v. 23, n. 3, p. 16–27, 1986. 39

——. The solution of large 0-1 integer programming problems encountered in automated cartography. **Operations Research**, v. 38, n. 5, p. 752–759, 1990. 39, 42, 46, 54

——. Expert systems and the map label placement problem. **Cartographica**, v. 28, n. 1, p. 1–9, 1991. 39