Ministério da
Ciência e Tecnologia

INPE-15388-TDI/1405

RELAXAÇÕES E MÉTODO DE DECOMPOSIÇÃO PARA ALGUNS PROBLEMAS DE LOCALIZAÇÃO DE FACILIDADES MODELADOS EM GRAFOS

Francisco de Assis Corrêa

Tese de Doutorado do Curso de Pós-Graduação em Computação Aplicada, orientada pelo Dr. Luiz Antonio Nogueira Lorena, aprovada em 3 de outubro de 2008

Registro do documento original:
<http://urlib.net/sid.inpe.br/mtc-m18@80/2008/08.28.19.58>

INPE
São José dos Campos
2008

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GB)
Serviço de Informação e Documentação (SID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3945-6911/6923
Fax: (012) 3945-6919
E-mail: pubtc@sid.inpe.br

Ministério da
Ciência e Tecnologia

INPE-15388-TDI/1405

RELAXAÇÕES E MÉTODO DE DECOMPOSIÇÃO PARA ALGUNS PROBLEMAS DE LOCALIZAÇÃO DE FACILIDADES MODELADOS EM GRAFOS

Francisco de Assis Corrêa

Tese de Doutorado do Curso de Pós-Graduação em Computação Aplicada, orientada pelo Dr. Luiz Antonio Nogueira Lorena, aprovada em 3 de outubro de 2008

Registro do documento original:
<http://urlib.net/sid.inpe.br/mtc-m18@80/2008/08.28.19.58>

INPE
São José dos Campos
2008

Dados Internacionais de Catalogação na Publicação (CIP)

C817r Corrêa, Francisco de Assis.
Relaxações e método de decomposição para alguns problemas de localização de facilidades modelados em grafos / Francisco de Assis Corrêa. – São José dos Campos: INPE, 2008.
134p. ; (INPE-15388-TDI/1405)

Tese (Doutorado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2008.

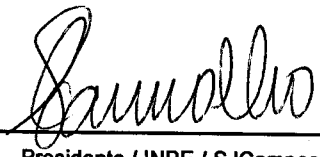
1. Relaxação Lagrangeana. 2. Relaxação Lagrangeana com clusters. 3. Algoritmo de geração de colunas. 4. Localização de facilidades. 5. Particionamento de grafos. I. Título.

CDU 681.5.015.23

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de Doutor(a) em

Computação Aplicada

Dr. Solon Venâncio de Carvalho

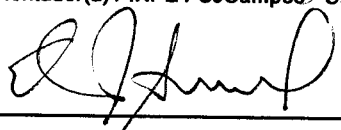


Presidente / INPE / SJCampos - SP

Dr. Luiz Antonio Nogueira Lorena

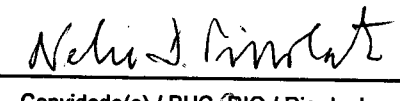
Orientador(a) / INPE / SJCampos - SP

Dr. Edson Luiz Franca Senne



Membro da Banca / UNESP/GUARA / Guaratinguetá - SP

Dr. Nelio Domingues Pizzolato



Convidado(a) / PUC - RIO / Rio de Janeiro - RJ

Dr. Reinaldo Morabito Neto

Convidado(a) / UFSCAR / São Carlos - SP

Aluno (a): Francisco de Assis Corrêa

São José dos Campos, 03 de outubro de 2008

A meus filhos e a minha esposa.

AGRADECIMENTOS

Primeiramente, agradeço a Deus, cujas bençãos permitiram esta caminhada.

A minha família pelo apoio e incentivo.

Agradeço o meu orientador, Prof. Luiz Antonio Nogueira Lorena, pelo apoio, competência, orientação, conhecimento transmitido e paciência demonstrados nesse período de convivência.

Ao Instituto Nacional de Pesquisas Espaciais - INPE, pela oportunidade de estudar nesta inspiradora instituição.

Aos professores do INPE, especialmente aos do Laboratório Associado de Matemática e Computação Aplicada - LAC, pela convivência amiga e pelos conhecimentos transmitidos.

À Pesquisadora Mônica Maria De Marchi, do Instituto de Estudos Avançados - IEAv, pelo incentivo para o início desta jornada.

Ao Prof. Edson Luiz França Senne, pelo apoio e pelas oportunidades de desenvolvimento de trabalhos conjuntos.

Aos meus amigos e colegas de curso, em especial Glaydston, Geraldo e Antônio por todo o suporte, incentivo e, principalmente, pelo intercâmbio de informações que deixaram o meu fardo mais leve.

Aos profissionais do Serviço de Pós-Graduação (SPG) e às secretárias do LAC pelo apoio e dedicação.

RESUMO

Apesar do grande avanço na área de *hardware* computacional e das melhores técnicas atuais para a resolução de problemas de otimização combinatória, nem sempre é possível a obtenção do ótimo global para alguns problemas práticos de localização de facilidades em um tempo computacional aceitável devido à classificação como *NP-hard* e ao porte do problema. Esta tese explora a representação de restrições de problemas em grafos e uma estratégia de particionamento para resolver problemas de localização de facilidades, usando relaxações e método de decomposição. São abordados dois problemas de localização de facilidades: o Problema de Localização de Facilidades não-Capacitado e o Problema Probabilístico de Localização-Alocação de Máxima Cobertura. Para os dois foram aplicados a Relaxação Lagrangeana com *Clusters* (LagClus) e um Algoritmo de Geração de Colunas. O primeiro problema foi modelado por meio de um grafo de conflitos. O segundo foi modelado por um grafo de cobertura, pois, devido às características do problema, gera uma quantidade menor de restrições relaxadas ou de acoplamento, permitindo a obtenção de melhores limitantes. A relaxação lagrangeana tradicional e a relaxação de Programação Linear também foram aplicadas a esses problemas, de forma a obter valores para comparar com os resultados da LagClus e da Geração de Colunas. O Problema Probabilístico de Localização-Alocação de Máxima Cobertura foi ainda resolvido usando o Algoritmo Genético Construtivo (AGC) e o *Clustering Search* (CS). Os testes computacionais em instâncias da literatura mostram resultados interessantes, e significativas conclusões são apresentadas.

RELAXATIONS AND DECOMPOSITION APPROACH FOR SOME FACILITY LOCATION PROBLEMS MODELED BY GRAPHS

ABSTRACT

Despite the great advances in computational equipment and the best known techniques for solving combinatorial optimization problems, it is not always possible to find the optimum solution to some practical facility location problems in a reasonable computational time, due to their size and classification issues. This thesis explores the representation of problem constraints in graphs and a graph partitioning strategy to solve facility location problems, using a relaxation and decomposition approach. Two problems will be considered: the Uncapacitated Facility Location Problem and the Queueing Maximal Covering Location-Allocation Model. For both problems the Lagrangean Relaxation with Clusters (LagClus) and a Column Generation Method have been applied. The first has been modeled by a conflict graph. The second has been modeled by a covering graph that produces a smaller quantity of relaxed or coupling constraints for this problem to obtain better bounds. Traditional lagrangean relaxation and linear programming relaxation have been applied to these problems in order to obtain values to compare with the results of the LagClus and Column Generation. The Queueing Maximal Covering Location-Allocation Model has also been solved using the Constructive Genetic Algorithm and the Clustering Search. The computational tests report interesting results for instances from the literature and significant conclusions are described.

SUMÁRIO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE ABREVIATURAS E SIGLAS

AGC	–	Algoritmo Genético Construtivo
CS	–	Clustering Search
DL	–	Dual Lagrangeano
DW	–	Decomposição Dantzig-Wolfe
GAP	–	*Generalized Assignment Problem*
GRASP	–	*Greedy Randomized Adaptive Search Procedure*
LagClus	–	Relaxação Lagrangeana com *Clusters*
LB	–	*Lower Bound*
LSCP	–	*Location Set Covering Problem*
MCLP	–	*Maximal Covering Location Problem*
OC	–	Otimização Combinatória
PL	–	Relaxação de Programação Linear
PM	–	Problema Mestre
PMCIVP	–	Problema do Máximo Conjunto Independente de Vértices com Pesos
PMP	–	Problema das p-Medianas
PMR	–	Problema Mestre Restrito
PPG	–	Problema de Particionamento de Grafos
PPLAMC	–	Problema Probabilístico de Localização-Alocação de Máxima Cobertura
RP	–	Relaxação do Problema (P)
UB	–	*Upper Bound*
UFLP	–	*Uncapacitated Facility Location Problem*

LISTA DE SÍMBOLOS

A	–	Matriz binária de coeficientes das restrições associadas às arestas entre *clusters*
B_k	–	Matriz binária de coeficientes das restrições do *clusters* k
DL	–	Problema dual lagrangeano
E	–	Conjunto de arestas de um grafo
E_k	–	Conjunto de arestas do subgrafo k
$\hat{E}$	–	Conjunto de arestas que conectam subgrafos
G	–	Representação de um grafo
J_k	–	Conjunto de pontos extremos do *cluster* k
L_uP	–	Relaxação lagrangeana do problema P
$LC_\mu P$	–	Relaxação lagrangeana com *clusters* do problema P
P	–	Problema de otimização
P_{DW}	–	Decomposição Dantzig-Wolfe do problema P
Q	–	Quantidade de restrições relaxadas
O_k	–	Restrições associadas ao subproblema k
K	–	Número de partições de um grafo
$\Re$	–	Conjunto de números reais
$\Re^+$	–	Conjunto de números reais não negativos
$\Re^n$	–	Conjunto de números reais de dimensão n
$\Re^n_+$	–	Vetor de números reais não negativos de dimensão n
u	–	Multiplicadores lagrangeanos usados na relaxação lagrangeana tradicional
V	–	Conjunto de vértices de um grafo
Z^n_+	–	Vetor de números inteiros não negativos de dimensão n
α	–	Vetor de variáveis duais associadas às restrições de acoplamento
β_k	–	Variável dual associada à k-ésima restrição de convexidade
λ_{jk}	–	Variáveis de decisão que correspondem aos pontos extremos $j \in J_k$
μ	–	Multiplicadores lagrangeanos associados com as linhas da matriz A_k, na LagClus
π	–	Tamanho do passo no algoritmo de subgradientes
$\|x\|$	–	Norma do vetor x

1 INTRODUÇÃO

Os problemas de otimização dividem-se em problemas com variáveis contínuas e com variáveis discretas, esses últimos denominados combinatoriais. Nos problemas contínuos, busca-se um conjunto de números reais ou mesmo uma função; nos problemas combinatoriais, busca-se um objeto de um conjunto finito, ou possivelmente infinito contável (PAPADIMITRIOU; STEIGLITZ, 1998).

Um problema de otimização pode ser definido como um conjunto de instâncias, onde cada uma é um par (F, c), sendo F um conjunto que representa o conjunto de soluções viáveis, e c a função custo, tal que

$$c : F \rightarrow \Re^1$$

O problema é encontrar um $f \in F$ para o qual $c(f) \leq c(y), \forall y \in F$. O ponto f é denominado solução ótima global (PAPADIMITRIOU; STEIGLITZ, 1998). Neste caso, foi considerado o problema de minimização. Para maximização, toma-se $c(f) \geq c(y), \forall y \in F$.

De forma resumida, otimização é a busca da melhor solução para um dado problema dentro de um conjunto finito ou infinito de possíveis soluções. Um problema de otimização é composto de uma função objetivo a ser otimizada e de restrições que definem o conjunto F. Uma função objetivo que apresenta um único ponto de mínimo é denominada de função unimodal. Quando uma função apresenta mais de um ponto de mínimo, ela é denominada de função multimodal.

Particularmente, a otimização combinatória (OC) caracteriza-se por apresentar um espaço de busca composto de combinações dos elementos de estruturas discretas, cujo objetivo é encontrar o mínimo ou o máximo global de um determinado problema, dado um conjunto de possíveis soluções. No entanto, para vários problemas de interesse, a busca da solução ótima por um processo enumerativo (busca exaustiva) torna-se inviável, devido ao tempo necessário para testar todo o conjunto. Para cada pequeno incremento na dimensão do problema, surge um grande número de possibilidades, gerando uma explosão combinatória de possíveis soluções e, conseqüentemente, uma explosão também no tempo computacional (BJORNDAL et al., 1995).

Os problemas de otimização que possuem funções multimodais podem ter uma complexidade tal que seja impraticável a obtenção do ótimo global em um tempo computacional aceitável (por meio de métodos exatos baseados em programação matemática). Como alternativa para solução desses tipos de problemas, várias técnicas de solução vêm sendo desenvolvidas ou aprimoradas, entre elas modelos matemáticos mais

ajustados, relaxações, heurísticas, metaheurísticas e métodos de decomposição.

As relaxações possuem a vantagem de definir limitantes para a solução ótima e podem 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. Entre as relaxações, destacam-se a de programação linear, a Lagrangeana (BEASLEY, 1993), a Surrogate e a Lagrangeana/Surrogate (NARCISO; LORENA, 1999) e a Lagrangeana com *clusters* (LagClus) (RIBEIRO, 2007).

Os métodos heurísticos e as metaheurísticas são alternativas para a busca de boas soluções sem a garantia da obtenção do ótimo global, porém em um tempo computacional reduzido.

Algoritmos de Geração de Colunas, Algoritmos *Branch-and-Price* e Algoritmos *Branch-and-Cut* são alguns métodos de decomposição atualmente estudados. Esses algoritmos aproveitam características do problema original para decompô-lo em problemas independentes (RIBEIRO, 2007).

A solução de um problema computacional inicia-se com a busca de uma estrutura de dados adequada a sua representação. Uma das mais utilizadas em otimização é a denominada grafo, definido por $G = (V, E)$, em que V é um conjunto de vértices e E um conjunto de pares de vértices chamados arestas. Os vértices podem, por exemplo, representar pontos de demanda de uma população e facilidades a serem instaladas em uma cidade, como hospitais, bancos e postos policiais; e as arestas podem representar as ligações entre esses dois elementos, que podem conter informações como tempo ou distância entre eles.

Um caso particular de grafo é o denominado grafo de conflitos, que representa relações lógicas entre variáveis binárias. Existe um vértice para cada variável binária e seu complemento, e uma aresta entre dois vértices quando no máximo uma das variáveis representadas pelos vértices podem ser iguais a um na solução ótima. As arestas de um grafo de conflitos são denominadas arestas de adjacências ou conflitos (ATAMTÜRK et al., 2000).

Outro caso particular de grafo é o denominado grafo de cobertura, que permite representar um problema de localização com cobertura em que os vértices representam as localizações candidatas, e as arestas, a cobertura de cada ponto de demanda (CORRÊA et al., 2008).

Uma estratégia usada para a solução de um problema de grande porte é a sua divisão em problemas menores (*dividir para conquistar*), com a mesma característica do problema original, resolvendo-os com a ajuda de *solvers* comerciais. Essa divisão pode ser conseguida com o particionamento do grafo, de forma a se obterem aglomerados (*clusters*) de vértices e

arestas. Busca-se com esse particionamento separá-lo em subgrafos. Ao remover as arestas que ligam os agrupamentos, obtêm-se partes com as mesmas características do problema original. O inconveniente dessa abordagem é que não há a garantia de que resolvendo-se os subproblemas obtém-se a solução para o problema original, pois trata-se de uma relaxação; porém, pode-se produzir um limitante de boa qualidade, o que torna o processo interessante de ser aplicado.

De modo geral, apesar de se usar uma estrutura adequada a cada tipo de problema, existe o inconveniente de nem sempre ser possível encontrar a solução ótima em um tempo computacional viável, devido à sua classificação como *NP-hard* (DASKIN, 1995) e ao porte do problema. Alguns problemas práticos permanecem desafiadores, mesmo com a utilização de *solvers* comerciais que contemplam as melhores técnicas atuais para a resolução de problemas de OC e com o grande avanço na área de *hardware* computacional.

Entre esses problemas de OC encontram-se os problemas de localização de facilidades, cuja idéia central reside na escolha de locais adequados para implantar facilidades (centros) para o atendimento de clientes (pontos de demanda), considerando-se que certos critérios de otimização devem ser atendidos. As facilidades são centros que prestam algum tipo de serviço, tais como fábricas, hospitais, bancos e escolas, que têm como clientes, respectivamente, depósitos, pacientes, correntistas e estudantes. Quando existe a necessidade de definir qual centro atenderá que pontos de demanda, tem-se um problema de localização-alocação.

A representação de um problema de OC em grafos e a estratégia de particionamento sugerida podem ser exploradas em relaxações e em métodos de decomposição. Estas, juntamente com a aplicação de algumas metaheurísticas em problemas de localização de facilidades, são os objetivos principais deste trabalho, cujas principais contribuições são:

- Aplicação da LagClus e do Algoritmo de Geração de Colunas à solução do Problema de Localização de Facilidades Não-capacitado (*Uncapacitated Facility Location Problem* - UFLP), utilizando a abordagem do grafo de conflitos.
- Resolução de um Problema Probabilístico de Localização-alocação de Máxima Cobertura (PPLAMC) com as metaheurísticas Algoritmo Genético Construtivo (AGC) e o *Clustering Search* (CS).
- Aplicação da LagClus e do Algoritmo de Geração de Colunas à solução de um Problema Probabilístico de Localização-alocação de Máxima Cobertura, utilizando a abordagem do grafo de cobertura.

Esta tese está organizada da seguinte forma:

- **CAPÍTULO 2**: Apresentação dos conceitos básicos sobre problemas de localização de facilidades, grafos (de conflitos e de cobertura), particionamento de grafos, relaxações, geração de colunas, Algoritmo Genético Construtivo e *Clustering Search* (CS).
- **CAPÍTULO 3**: Apresentação do problema de localização de facilidades não-capacitado (UFLP), da aplicação da relaxação lagrangeana e, usando-se o grafo de conflitos, da aplicação da LagClus e do algoritmo de geração de colunas a esse problema. Apresentação dos resultados computacionais obtidos a partir de instâncias obtidas na *OR-Library* e em outras com grande *gap* de dualidade e de difícil solução para a relaxação de programação linear.
- **CAPÍTULO 4**: Apresentação de um Problema Probabilístico de Localização-alocação de Máxima Cobertura (PPLAMC) e sua formulação e as adaptações necessárias à aplicação da LagClus e do Algoritmo de Geração de Colunas, usando-se o grafo de cobertura. Também são apresentadas as abordagens do Algoritmo Genético Construtivo e *Clustering Search*. Apresentação dos resultados computacionais aplicados em instâncias com até 818 vértices.
- **CAPÍTULO 5**: Apresentação do resumo das principais contribuições, conclusões e sugestões para trabalhos futuros.

2 PROBLEMAS DE LOCALIZAÇÃO DE FACILIDADES E ALGUNS MÉTODOS DE SOLUÇÃO

A localização de facilidades é um problema de otimização combinatória, que consiste na escolha de locais adequados para implantar facilidades (centros) para o atendimento de clientes (pontos de demanda), considerando-se que certos critérios de otimização devem ser atendidos.

Os problemas de localização de facilidades discutidos neste trabalho são *NP-Hard* (CORNUÉJOLS et al., 1990) e (DASKIN, 1995). Por isso, buscam-se técnicas alternativas para a solução desse tipo de problema, algumas das quais serão abordadas neste capítulo: metaheurísticas, relaxações e decomposição. Além disto, serão descritos conceitos de grafos e particionamento de grafos, que juntamente com as relaxações e decomposição formam a base para a Relaxação Lagrangeana com *Clusters* (LagClus) e do Algoritmo de Geração de Colunas, empregados na solução de alguns problemas de localização de facilidades, motivo deste trabalho.

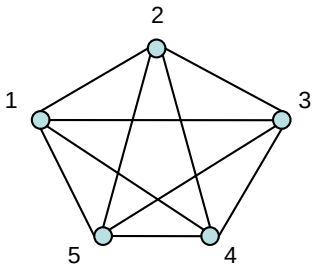
Ainda como alternativa, podem-se considerar as heurísticas. Porém, por serem definidas especificamente para um determinado problema, são tratadas nos capítulos respectivos.

Alguns problemas clássicos de localização de facilidades são também discutidos neste capítulo.

2.1 Representação em grafos

Muitos problemas envolvendo situações reais podem ser convenientemente representados por meio de um diagrama que consiste em um conjunto de pontos e um conjunto de linhas ligando alguns ou todos os pares desses pontos. Por exemplo, os pontos do diagrama podem representar diferentes cidades de um país, e a linha ligando dois pontos pode indicar que existe um serviço aéreo direto entre as cidades representadas por esses pontos. Se houver o serviço aéreo apenas do ponto A para o B, uma seta com início em A e final em B será desenhada entre esses pontos, e não de B para A. A abstração matemática de tais estruturas envolvendo pontos e linhas motiva o conceito de grafos (BALAKRISHNAN, 1997). Mais detalhes sobre este assunto podem ser obtidos em (CHRISTOFIDES, 1975), (WILSON; WATKINS, 1990), (CHARTRAND; OELLERMANN, 1992), (NETTO, 1996) e (WEST, 2000).

Um grafo não orientado $G = (V,E)$ consiste em um conjunto não vazio V de vértices e um conjunto E de pares não ordenados de vértices distintos de V, ou seja $E = \{(u, v) : u, v \in V, u \neq v\}$. Cada par de vértices em E é uma aresta em G. Um grafo G é dito ponderado se a cada vértice e/ou aresta estiver associado um número real, denominado peso.



	1	2	3	4	5
1	0	5	6	3	2
2	5	0	5	7	6
3	6	5	0	4	6
4	3	7	4	0	1
5	2	6	6	1	0

Figura 2.1 - Grafo G = (V, E) e um possível conjunto de pesos nas arestas.

Vários problemas de Otimização Combinatória podem ser modelados sobre grafos. O grafo da Figura 2.1, se ponderado pelo possível conjunto de pesos ao lado, pode representar o Problema das p-Medianas (HAKIMI, 1964), em que os vértices são os pontos de demanda e as potenciais facilidades a serem abertas (considerando-se que cada ponto de demanda é um possível ponto para a localização de uma facilidade), e os pesos nas arestas podem significar a distância entre esses pontos. O Problema das p-Medianas será discutido ainda neste Capítulo.

2.1.1 Grafo de conflitos

Conforme mencionado, um grafo de conflitos representa relações lógicas entre variáveis binárias. Existe um vértice para cada variável binária e seu complemento, e uma aresta entre dois vértices quando no máximo uma das variáveis representadas pelos vértices podem ser iguais a um em uma solução factível. As arestas de um grafo de conflitos são denominadas arestas de adjacências ou conflitos. As quatro possíveis relações lógicas entre duas variáveis binárias x_i e x_j são (ATAMTÜRK et al., 2000):

$$x_i = 1 \Rightarrow x_j = 0 \Leftrightarrow x_i + x_j \leq 1$$

$$x_i = 0 \Rightarrow x_j = 0 \Leftrightarrow (1 - x_i) + x_j \leq 1$$

$$x_i = 1 \Rightarrow x_j = 1 \Leftrightarrow x_i + (1 - x_j) \leq 1$$

$$x_i = 0 \Rightarrow x_j = 1 \Leftrightarrow (1 - x_i) + (1 - x_j) \leq 1$$

A Figura 2.2 é um exemplo de um grafo de conflitos que representa as três restrições descritas a seguir. As linhas duplas do grafo mostram o conflito entre uma variável e seu complemento:

$$x_i + (1 - x_j) \leq 1$$

$$x_i + x_k \leq 1$$

$$(1 - x_j) + x_k \leq 1$$

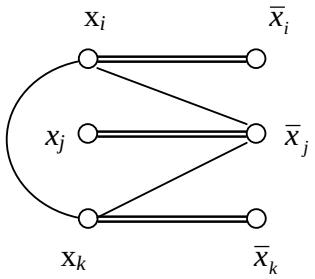


Figura 2.2 - Grafo de conflitos.

Fonte: Adaptada de Atamtürk et al. (2000).

Vários problemas encontrados na literatura utilizam grafo de conflitos para a representação de restrições de um problema específico. Entre eles, podem-se mencionar: *airline crew-scheduling* (HOFFMAN; PADBERG, 1993), *scheduling of jobs* (IRANI; LEUNG, 2003), *timetabling* (BURKE et al., 1994), bioinformática (BAFNA; BANSAL, 2004), Problema de Estivagem de Unidades de Celulose (RIBEIRO; LORENA, 2008b), Problema de Carregamento de Paletes ao Produtor (RIBEIRO; LORENA, 2007), o Problema de Rotulação Cartográfica de Pontos (RIBEIRO; LORENA, 2008a), e *Uncapacitated Facility Location Problem* (CORRÊA et al., 2006a; CORRÊA et al., 2006b; CORRÊA et al., 2008).

2.1.2 Grafo de cobertura

Considere-se um problema de localização de facilidades com cobertura. Considere-se ainda $i \in I$ e $j \in N_i$, em que I é o conjunto dos pontos de demanda a serem alocados e N_i é o conjunto de localizações candidatas que estão dentro de uma distância padrão do nó i ou o conjunto de localizações candidatas que podem ser alcançadas do nó i em um certo tempo. Seja $N = \cup_{i \in I} N_i$.

Define-se o grafo de cobertura $G = (V, E)$, em que $V = N$ e $E = \{(u, v) : u, v \in N_i, \forall i \in I\}$.

Como exemplo, considere-se $I = \{1, ..., 5\}$, $N = \{1, ..., 5\}$ e os valores mostrados na Tabela 2.1.

Tem-se $N_1 = \{1, 2, 4, 5\}$, $N_2 = \{1, 2, 3\}$, $N_3 = \{2, 3, 4\}$, $N_4 = \{1, 3, 4, 5\}$ e $N_5 = \{1, 4, 5\}$.

Tabela 2.1 - Dados de cobertura.

Pontos de demanda	Cobertura
1	1, 2, 4, 5
2	1, 2, 3
3	2, 3, 4
4	1, 3, 4, 5
5	1, 4, 5

A Figura 2.3 mostra as cliques do ponto de demanda 1 (N_1) e do ponto de demanda 3 (N_3). A Figura 2.4 mostra o grafo de cobertura (cobertura de cliques) completo para este exemplo:

Figura 2.3 - Cliques (cobertura) dos pontos de demanda 1 (N1) e 3 (N3).

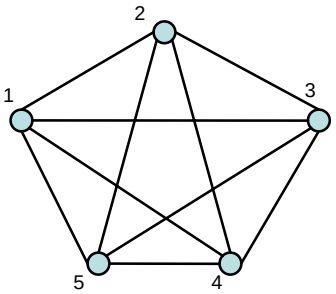


Figura 2.4 - Grafo de cobertura completo.

O grafo de cobertura foi utilizado para resolver o Problema Probabilístico de Localização-alocação de Máxima Cobertura, que será tratado no Capítulo 4.

2.1.3 Particionamento de grafos

Dado um número K e um grafo $G = (V, E)$, em que V é um conjunto de vértices com pesos, e E um conjunto de arestas também com pesos, o problema de particionamento de grafos é o de encontrar K subconjuntos $V_1, V_2, ..., V_K$ de V, tal que:

a) $\cup_{i=1}^{K} V_i = V$ e $V_i \cap V_j = \emptyset, \quad \forall i \neq j$.

b) $W(i) \approx W/K, \ i = 1, 2, ..., K$, em que $W(i)$ e W representam, respectivamente, as somas dos pesos dos vértices pertencentes a V_i e V.

c) A soma dos pesos das arestas que conectam os subconjuntos deve ser mínima.

Todo conjunto $\{V_i \subseteq V : 1 \leq i \leq K\}$ é chamado de uma partição se satisfaz a condição a), ou seja, cada V_i é uma parte de um K-particionamento. O particionamento que satisfaz b) é dito balanceado. De modo geral, o objetivo consiste então em dividir G em K partes, de forma a minimizar a quantidade de cortes nas arestas e reduzir o desbalanceamento dos pesos nessas partes.

O problema de particionamento de grafos (PPG) é classificado como *NP-hard* (GAREY et al., 1976). Por isso, algoritmos capazes de encontrar a solução ótima, como o de Karisch et al. (2000), são indicados para grafos pequenos (FJÄLLSTRÖM, 1998). Para problemas de grande porte recorre-se à implementação de algoritmos heurísticos.

Entre os algoritmos heurísticos encontrados na literatura, podem ser citados: algoritmo de Kernighan e Lin (KERNIGHAN; LIN, 1970), bissecção recursiva de Simon (SIMON, 1991), Karypis e Kumar (KARYPIS; KUMAR, 1998a; KARYPIS; KUMAR, 1998b), algoritmo paralelo de Gilbert e Zmijewski (GILBERT; ZMIJEWSKI, 1987) e os algoritmos de Johnson et al. (JOHNSON et al., 1989), de Rolland et al. (ROLLAND et al., 1996) e de Bui e Moon (BUI; MOON, 1996), baseados em, respectivamente, *Simulated Annealing*, Busca Tabu e Algoritmo Genético.

Alguns trabalhos merecem destaque em particionamento de grafos: o de Simon (SIMON, 1991) e os de Karypis e Kumar (KARYPIS; KUMAR, 1998a; KARYPIS; KUMAR, 1998b). O primeiro, por ser um dos trabalhos mais citados na literatura e os de Karypis e Kumar, por criarem o aplicativo METIS usado no particionamento dos grafos de conflitos e de cobertura empregados neste trabalho.

O primeiro (SIMON, 1991) inicia-se localizando dois vértices que possuem o maior caminho entre si. Um deles é selecionado e, usando-se busca em largura, a distância desse vértice em

relação aos demais é determinada. Os vértices são ordenados em relação a essa distância, e o conjunto ordenado é dividido em dois subconjuntos de mesmo tamanho e assim sucessivamente.

Os algoritmos propostos por Karypis e Kumar (KARYPIS; KUMAR, 1998a; KARYPIS; KUMAR, 1998b) empregam a técnica da bissecção recursiva em multi-níveis, que tem se tornado o método padrão para particionar grafos grandes e irregulares. Além disso, apresentam soluções de boa qualidade em tempo computacional baixo .

Os algoritmos multi-níveis, em vez de tentarem realizar o particionamento do grafo original diretamente, fazem sucessivas aproximações desse grafo. Em cada uma delas, um novo grafo é gerado, de tamanho menor que o anterior. Essa fase é chamada de *Coarsening*. O processo continua até que se gere um grafo pequeno, com algumas dezenas de vértices. Então, por ser de pequeno porte, esse grafo pode ser particionado por meio de heurísticas eficientes, gerando boas soluções. O passo final desses algoritmos é propagar o particionamento do menor grafo gerado em direção ao problema original. Essa propagação é realizada por meio de sucessivas aproximações do particionamento, seguidas por métodos de refinamento. Esta fase é denominada *Uncoarsening* com refinamento.

O METIS é capaz de realizar particionamentos de boa qualidade em questão de segundos, mesmo em grafos com milhares de vértices. Mais detalhes sobre este assunto podem ser obtidos em Ribeiro (2007) e na página *web* do METIS e seu manual, disponível em (`http://glaros.dtc.umn.edu/gkhome/metis/metis/download`).

Outras referências, programas e *links* para particionamento de grafos podem ser encontrados em: (`http://www.ace.ual.es/~cgil/grafos/Graph_Partitioning.html`).

2.2 Problemas de localização de facilidades

Nos problemas de localização de facilidades, uma grande variedade de respostas pode ser dada ao simples questionamento sobre qual será a melhor configuração para localizar algum tipo de serviço. Isso dependerá dos critérios que devem ser adotados para este estudo.

Um tipo de problema bastante estudado na literatura é o denominado *Problema de Localização de Facilidades*, que envolve custos fixos para a localização de facilidades e custos de produção, de transporte e distribuição para satisfazer a demanda de determinadas regiões. Quando cada potencial facilidade tem uma capacidade, é chamado de *Problema de Localização de Facilidades Capacitado*, ou, do inglês, *Capacitated Facility Location Problem*. Quando não existe essa restrição, de *Problema de Localização de*

Facilidades Não-Capacitado, ou, do inglês, *Uncapacitated Facility Location Problem* (UFLP) (CORNUÉJOLS et al., 1990).

Pode-se, ainda, estar interessado em instalar uma rede de postos de saúde, de maneira que a sua localização minimize a distância ou o tempo de deslocamento dos usuários às facilidades instaladas, ou estar interessado em que o usuário esteja dentro de um determinado tempo ou distância dessas facilidades. A primeira abordagem corresponde ao que é conhecido na literatura como Problema das p-Medianas; e a segunda, como Problemas de Cobertura.

Uma boa revisão sobre problemas de localização pode ser encontrada em (HALE; MOBERG, 2003), (SERRA; MARIANOV, 2004), (GALVÃO, 2004) e (REVELLE; EISELT, 2005). Uma abordagem mais completa do assunto pode ser vista em (CORNUÉJOLS et al., 1990) e (DASKIN, 1995).

2.2.1 Problema de Localização de Facilidades Não-Capacitado

Este problema tem como objetivo localizar facilidades para satisfazer a demanda de determinadas regiões, de forma a minimizar o custo total do sistema, que envolve custos fixos para essa localização e custos de produção, de transporte e de distribuição.

Entre as formulações mencionadas na literatura para problemas de localização, o UFLP destaca-se pela grande aplicabilidade em problemas reais de decisão, mesmo em áreas não relacionadas à de localização de facilidades, como o particionamento de uma base de dados e a alocação das partições em um sistema de computação distribuída (PIRKUL, 1986) e roteamento de veículos (SINGH, 2008).

Vários métodos para a solução do UFLP têm sido elaborados por pesquisadores nas últimas décadas. A heurística gulosa desenvolvida por Kuehn e Hamburguer (KUEHN; HAMBURGUER, 1963) forneceu a base para o desenvolvimento e aplicação de vários algoritmos construtivos, métodos de busca local e heurísticas sofisticadas para esse problema. Existem vários algoritmos exatos, como em Cornuéjols et al. (1977) e Körkel (1989). Erlenkotter (1978) desenvolveu uma abordagem dual para o UFLP. Embora existam algoritmos exatos para a solução do problema, a busca por algoritmos alternativos (heurísticas, metaheurísticas e relaxações) é a escolha natural para a solução de instâncias de larga escala, devido à natureza *NP-Hard* do UFLP (CORNUÉJOLS et al., 1990). Uma heurística efetiva e bastante usada é a heurística lagrangeana (BEASLEY, 1993) que é baseada na relaxação lagrangeana com o algoritmo de otimização por subgradientes (DASKIN, 1995). Corrêa et al. (2006a) e Corrêa et al. (2006b) aplicaram, respectivamente, a Relaxação Lagrangeana com *Clusters* (LagClus) e um algoritmo de geração de colunas

ao UFLP, técnicas que serão discutidas ainda neste capítulo. Várias soluções utilizando metaheurísticas foram aplicadas, tais como *Simulated Annealing* (ALVES; ALMEIDA, 1992), Algoritmos Genéticos (KRATICA et al., 2001) e Busca Tabu (MICHEL; HENTENRYCK, 2003), (GHOSH, 2003), (SUN, 2006). Alguns trabalhos usaram a abordagem de redes neurais artificiais (GEN et al., 1996) (VAITHYANATHAN et al., 1996). Resende e Werneck (2006) adaptaram ao UFLP uma heurística híbrida que havia sido desenvolvida para a solução do Problema das p-Medianas. Essa heurística é chamada híbrida, porque combina elementos de várias metaheurísticas. Cornuéjols et al. (1990) apresentam um bom resumo histórico, revisão de aplicações e métodos de solução para o UFLP.

Uma modelagem matemática do problema de localização de facilidades não-capacitado (UFLP) é obtida se consideradas as variáveis a seguir descritas (CORNUÉJOLS et al., 1990). Considera-se $i \in I$ e $j \in J$, em que I é o conjunto dos pontos de demanda a serem alocados e J é o conjunto de localizações candidatas. Seja $y_j = 1$, se uma facilidade é localizada no vértice j e $y_j = 0$, caso contrário; seja $x_{ij} = 1$, se o ponto de demanda i for alocado à facilidade j, $x_{ij} = 0$, caso contrário. O parâmetro c_{ij} define o custo de servir o cliente i pela facilidade j e o parâmetro f_j define o custo fixo de se abrir uma facilidade em j. Então, o *UFLP* pode ser formalmente definido como:

$$v(UFLP) = \min \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + \sum_{j \in J} f_j y_j \tag{2.1}$$

Sujeito a

$$\sum_{j \in J} x_{ij} = 1 \quad \forall i \in I \tag{2.2}$$

$$x_{ij} \leq y_j \quad \forall i \in I,\ \forall j \in J \tag{2.3}$$

$$x_{ij} \in \{0,1\};\ y_j \in \{0,1\} \quad \forall i \in I,\ \forall j \in J \tag{2.4}$$

A função objetivo 2.1 minimiza o custo total do sistema. As restrições 2.2 impõem que cada ponto de demanda seja alocado a somente um centro. As restrições 2.3 definem que somente é possível alocar um ponto de demanda i a um centro j se houver um centro em j. As restrições 2.4 definem as condições de integralidade.

Para o problema capacitado é acrescentada a restrição 2.5, em que a_i define a demanda do ponto i e κ_j, a capacidade da facilidade na posição candidada j:

$$\sum_{i \in I} a_i x_{ij} \leq \kappa_j \quad \forall j \in J \tag{2.5}$$

2.2.2 Problemas de Localização com Coberturas

Uma maneira usual de definir a qualidade de serviços a serem prestados a usuários é considerar a distância (ou o tempo) entre pontos de demanda e a facilidade à qual eles são alocados. Normalmente, a alocação de demandas é feita à facilidade mais próxima. Se a distância ou o tempo entre eles estiver abaixo ou acima de um determinado valor crítico, o serviço será considerado, respectivamente, adequado ou inadequado (DASKIN, 1995).

Quando uma facilidade atende um ponto de demanda, diz-se que este último está coberto pelo primeiro. Em termos gerais, um ponto de demanda é dito coberto se existe pelo menos uma facilidade dentro de um raio de cobertura definido por uma distância ou um tempo de deslocamento previamente especificado. Otimizar a distância ou o tempo de deslocamento é o objetivo de vários modelos existentes em problemas de localização com cobertura.

Entretanto, a maneira de tratar a cobertura dos pontos de demanda depende dos objetivos buscados pelos tomadores de decisão. Em aplicações do setor privado, busca-se maximizar o lucro ou vencer a concorrência em ambientes competitivos, enquanto no setor público busca-se minimizar os custos ou maximizar a universalidade de atendimentos, a eficiência e a igualdade na distribuição dos serviços (MARIANOV, 2003). Esses últimos são difíceis de medir e são, normalmente, tratados pela minimização dos custos de localização e de operação necessários à cobertura total dos serviços ou pela busca da máxima cobertura, no caso de recursos restritos. Estes dois casos são vistos na literatura como, respectivamente, cobertura de conjuntos (*Location Set Covering Problem* - LSCP) e de máxima cobertura (*Maximal Covering Location Problem* - MCLP) (MARIANOV, 2003), com as formulações mostradas a seguir.

2.2.2.1 Cobertura de Conjuntos - LSCP

Este modelo busca localizar um número mínimo de centros necessários para obter a cobertura obrigatória de todas as demandas. Em outras palavras, toda demanda tem pelo menos um centro localizado dentro de uma distância ou tempo padrão.

O LSCP tem também uma grande importância prática, pois é usado para modelar uma

uma grande variedade de aplicações. Entre elas, podem-se citar: preservação ambiental (MARIANOV et al., 2008), escalonamento de tripulação de vôo (MEDARD; SAWHNEY, 2007) e o problema de loteria (JANS; DEGRAEVE, 2008). Boas referências para aplicações podem ser encontradas em Ceria et al. (1998), Ceria et al. (1997).

Vários métodos foram desenvolvidos para a solução do LSCP. Algoritmos exatos podem ser encontrados em Fisher e Kedia (1998), Beasley e Jørnsten (1992), Balas e Carrera (1996), Caprara e Toth (2000). Este último compara diferentes algoritmos exatos para o LSCP. Entretanto, devido à natureza *NP-hard* deste problema, buscam-se algoritmos alternativos para a solução de instâncias de larga escala, como as heurísticas de Monfroglio (1998), Brotcorne et al. (2002), Lan et al. (2007) e outras baseadas na relaxação lagrangeana (BEASLEY, 1990) e relaxação surrogate (LORENA; LOPES, 1994). Huisman (2007) aplicou um algoritmo de geração de colunas a este problema. Várias soluções utilizando metaheurísticas também foram desenvolvidas, tais como GRASP (BAUTISTA, 2007), Algoritmo Genético (LORENA; LOPES, 1997) e *Simulated Annealing* (JACOBS; BRUSCO, 1995).

Para definir formalmente o LSCP, as localizações são representadas pelas variáveis binárias x_j, com $x_j = 1$, se o ponto j for selecionado, $j = 0$, caso contrário.

$$v(LSCP) = Min \sum_{j \in J} x_j \tag{2.6}$$

Sujeito a

$$\sum_{j \in N_i} x_j \geq 1 \quad \forall i \in I \tag{2.7}$$

$$x_j \in \{0, 1\} \quad \forall j \in J \tag{2.8}$$

Em que

J = conjunto de pontos candidatos à facilidades (indexados por j).

I = conjunto de pontos de demanda (indexados por i).

$N_i = \{j | d_{ij} \leq S\}$ com d_{ij} = menor distância da localização potencial da facilidade j ao ponto de demanda i, e S = distância padrão de cobertura. N_i é o conjunto de todas as

potenciais posições candidatas à facilidades que estão dentro da distância padrão S do ponto de demanda i.

A função objetivo 2.6 minimiza a quantidade de facilidades necessárias. As restrições 2.7 definem que um ponto de demanda deve estar coberto por pelo menos um centro localizado dentro da distância padrão S. A natureza binária das variáveis é dada pela restrição 2.8.

2.2.2.2 Máxima Cobertura - MCLP

O Problema de Localização de Máxima Cobertura (MCLP) tem sido bastante tratado na literatura desde a sua formulação feita por Church e ReVelle (1974) e tem origem na necessidade de se obter a configuração ideal para localizar facilidades que atendam o maior número de indivíduos de uma população, considerada uma dada distância ou um tempo padrão do ponto de demanda. Não se busca com este modelo atender toda a população, mas oferecer o máximo de atendimento considerando os recursos disponíveis. Vários modelos aplicados a uma grande faixa de problemas, nos setores público e privado, são extensões dessa formulação, adaptados para melhor representar a realidade. Uma dessas extensões será vista no Capítulo 4. As aplicações variam de sistemas de emergência (EATON et al., 1986; CURRENT; O'KELLY, 1992), serviços hierárquicos de saúde (MOORE; REVELLE, 1982), controle de poluição do ar (HOUGLAND; STEPHENS, 1976), a sistemas congestionados (MARIANOV; SERRA, 1998; MARIANOV; SERRA, 2001). Os métodos de solução para o MCLP incluem relaxação de programação linear (CHURCH; REVELLE, 1974), heurísticas gulosas (DASKIN, 1995), relaxação lagrangeana (GALVÃO; REVELLE, 1996), relaxação lagrangeana/surrogate (LORENA; PEREIRA, 2002), geração de colunas (PEREIRA et al., 2007), entre outros métodos. Considerável revisão deste tema pode ser encontrada em Chung (1986), Hale e Moberg (2003), Serra e Marianov (2004), Galvão (2004).

Formalmente, as localizações são representadas pelas variáveis binárias y_i, com $y_i = 1$, se o nó i estiver coberto, e $y_i = 0$, caso contrário. Sendo a_i a demanda do ponto i, S a distância padrão de cobertura e p a quantidade de facilidades a serem abertas, tem-se:

$$v(MCLP) = Max \sum_{i \in I} a_i y_i \tag{2.9}$$

Sujeito a

$$y_i \leq \sum_{j \in N_i} x_j \quad \forall i \in I \tag{2.10}$$

$$\sum_{j \in J} x_j = p \tag{2.11}$$

$$x_j, y_i \in \{0, 1\} \quad \forall j \in J, \quad \forall i \in I \tag{2.12}$$

Em que, I, J, N_i e x_j são os conjuntos definidos para LSCP. A função objetivo 2.9 maximiza a demanda dos nós cobertos. As restrições 2.10 definem que a demanda no nó i estará coberta se existir pelo menos uma facilidade localizada dentro de um tempo ou distância padrão. A restrição 2.11 define o número total de facilidades a serem abertas. As restrições 2.12 tratam das condições de integralidade.

2.2.3 Problema das p-Medianas

A busca de p-medianas é um problema clássico de localização. Consiste em localizar p facilidades ou recursos (medianas), de forma a minimizar a soma das distâncias de cada vértice à sua facilidade mais próxima, estando sujeito a algumas restrições, tais como, cada vértice deve ser atendido por apenas uma facilidade instalada e exatamente p vértices devem ser escolhidos para a instalação das medianas.

O Problema das p-Medianas (*p-Median Problem* - PMP) apresenta um grande número de aplicações na literatura para a área de localização. A primeira foi a determinação de centros de comutação em redes de comunicações, feita por Hakimi (1964), Hakimi (1965). Também é aplicada em roteamento de veículos (KOSKOSIDIS; POWELL, 1992), particionamento de um território em distritos eleitorais (*political districting problems*) (BOZKAYA et al., 2003), projeto de redes de computadores (PIRKUL, 1987), modelo de localização dinâmica, em que uma facilidade pode estar fechada em qualquer período do horizonte de planejamento, porém, em um dado período, exatamente p facilidades devem estar em operação (GALVÃO; SANTIBANEZ-GONZALEZ, 1992), entre outras.

Os métodos de solução para o Problema das p-Medianas incluem algoritmos baseados no método *branch-and-bound* (JÄRVINEN et al., 1972), relaxação lagrangeana (CORNUÉJOLS et al., 1977), heurística lagrangeana/surrogate (SENNE; LORENA, 2000), geração de colunas (LORENA; SENNE, 2004; PEREIRA, 2005; SENNE et al., 2005). Métodos heurísticos podem ser encontrados em Teitz e Bart (1968), Pirkul (1987) e Lorena e Senne (2003). Várias soluções utilizando metaheurísticas também foram desenvolvidas, tais como GRASP (RESENDE; WERNECK, 2002), *Simulated Annealing* (CHIYOSHI; GALVÃO, 2000), Algoritmo Genético (ALP et al., 2003), Algoritmo Genético Construtivo (LORENA; FURTADO, 2001), Busca Tabu (BOZKAYA et al., 2003), VNS (FLESZAR; HINDI, 2008), *Clustering Search*

(CHAVES et al., 2008). Considerável revisão deste tema pode ser encontrada em Galvão (2004), Mladenovic et al. (2007).

O Problema das p-Medianas pode ser formulado como um problema de programação linear inteira binária. Formalmente, as alocações são representadas pelas variáveis binárias x_{ij}, tal que $i, j \in N$, em que $N = 1,..,n$ é o conjunto dos pontos de demanda a serem alocados. Considera-se $x_{ij} = 1$, se o ponto de demanda i for alocado à mediana j, $x_{ij} = 0$, caso contrário; $x_{jj} = 1$ se o nó j é uma mediana e $x_{jj} = 0$, caso contrário.

Sendo assim, este problema possui a formulação descrita a seguir:

$$v(PMP) = Min \sum_{i \in N} \sum_{j \in N} d_{ij} x_{ij} \tag{2.13}$$

Sujeito a

$$\sum_{i \in N} x_{ij} = 1; \quad j \in N \tag{2.14}$$

$$x_{ij} \leq x_{jj}; \quad \forall i, j \in N \tag{2.15}$$

$$\sum_{j \in N} x_{jj} = p \tag{2.16}$$

$$x_{ij} \in \{0, 1\}; \quad \forall i, j \in N \tag{2.17}$$

Em que:

- $[d_{ij}]_{n \times n}$ é uma matriz simétrica de custos (distâncias), com $d_{ii} = 0$, $\forall i \in N$.
- $[x_{ij}]_{n \times n}$ é a matriz de alocação.
- p é o número de medianas;
- n é o número de pontos de demanda.

As restrições 2.14 garantem que cada nó i seja alocado a apenas uma mediana j e as restrições 2.15 definem que essa alocação somente será feita se existir uma mediana em j.

A restrição 2.16 define a quantidade de medianas a serem localizadas e as restrições 2.17 tratam das condições de integralidade.

2.3 Relaxação

Considere o seguinte problema de programação inteira, escrito em notação matricial, referido como problema (P), em que x é o vetor de variáveis de decisão, c é o vetor de custos, b e e são vetores de disponibilidades de recursos e A e D são matrizes de coeficientes.

$$v(P) = Min\ cx \tag{2.18}$$

Sujeito a

$$Ax \leq b \tag{2.19}$$

$$Dx \leq e \tag{2.20}$$

$$x \in \{0, 1\} \tag{2.21}$$

Uma relaxação do problema (P) é um problema de otimização, (RP), que possui as seguintes propriedades: (i) o conjunto de soluções viáveis de (P) é um subconjunto das soluções viáveis de (RP), e (ii) considerando problemas de minimização, o valor da função objetivo de (P) pode ser maior que o correspondente valor de (RP), isto é, $v(P) \geq v(RP)$ (WOLSEY, 1998).

Como foi relatado, os problemas de otimização podem ter uma complexidade tal que seja impraticável a obtenção do ótimo global por meio de métodos exatos em um tempo computacional aceitável. Uma estratégia para resolver (P) consiste em resolver uma seqüência de problemas mais "fáceis", obtidos a partir de relaxações do problema original, de forma a se obter limites para a solução ótima de (P) a partir da solução desses problemas. É nesse contexto que as diferentes relaxações têm sido desenvolvidas ao longo do tempo. Pode-se considerar, sem perda de generalidade, que (P) é um problema viável e que o conjunto de soluções viáveis de cada problema relaxado obtido de (P) é um conjunto finito (ESPEJO; GALVÃO, 2002).

Para garantir a optimalidade de uma solução x* de (P) basta encontrar um limitante inferior (limitante dual) $\underline{x}$, tal que $v(\underline{x}) \leq v(x^*)$ e um limitante superior (limitante primal)

$\bar{x}$, tal que $v(\bar{x}) \geq v(x^*)$, de forma que $v(\underline{x}) = v(\bar{x})$. A optimalidade de uma solução pode ser buscada com um algoritmo que consiga encontrar uma ordem crescente de limitantes duais $v(\underline{x}_1) < v(\underline{x}_2) < ... < v(\underline{x}_d)$ e uma ordem decrescente de limitantes primais $v(\bar{x}_1) > v(\bar{x}_2) > ... > v(\bar{x}_p)$, até que $v(\bar{x}_p) - v(\underline{x}_d) < \varepsilon$. Dependendo do valor de ε (por exemplo, $\varepsilon < 1$, para problemas com coeficientes inteiros), pode-se garantir a optimalidade, que será o valor de $v(\bar{x}_p)$.

O maior desafio de uma relaxação é o de encontrar bons limitantes duais. Uma boa abordagem para esse desafio é a relaxação lagrangeana com a otimização do limitante dual obtida por meio do algoritmo de subgradientes (HELD; KARP, 1970; PARKER; RARDIN, 1988; PIRKUL; SCHILLING, 1991; LORENA; SENNE, 2003) que será vista a seguir.

2.3.1 Relaxação Lagrangeana

Define-se a relaxação lagrangena do problema (P) com respeito ao conjunto de restrições $Ax \leq b$ acrescentando-se um vetor de multiplicadores de Lagrange u com sinal apropriado a esse conjunto de restrições, incorporando o produto obtido à função objetivo. A relaxação lagrangeana (L_uP) do problema (P) é dada por:

$$v(L_uP) = Min\ cx + u(Ax - b) \tag{2.22}$$

Sujeito a

$$Dx \leq e \tag{2.23}$$

$$x \in \{0, 1\} \tag{2.24}$$

As restrições escolhidas $Ax \leq b$ são as restrições que permitem que o problema (P) seja mais "facilmente resolvido", caso sejam removidas.

O problema (L_uP) é um problema em x, resolvido para um dado vetor fixo $u \geq 0$, escolhido para garantir que $v(L_uP) \leq v(P)$. Se as restrições forem do tipo $Ax \geq b$, os multiplicadores devem ser não positivos para garantir $v(L_uP) \leq v(P)$. O sinal de u será irrestrito se $Ax = b$. A qualidade do limite gerado depende do valor de u e encontrar bons limites por meio dessa relaxação depende de se encontrar um conjunto de multiplicadores u que resolve o chamado dual lagrangeano *(DL)*, dado por:

$$\underset{u\geq 0}{Max} \left\{ \begin{array}{l} Min\ cx + u(Ax - b) \\ sujeito\ a\ Dx \leq e \\ x \in \{0,1\} \end{array} \right\} \tag{2.25}$$

O ideal é encontrar o valor ótimo do dual lagrangeano igual ao valor ótimo do problema original (P). Se esses valores não são iguais, diz-se, então, que existe um *gap* de dualidade, cujo valor é dado pela diferença entre os dois valores ótimos.

A função $v(L_uP)$ é linear por partes e côncava (PARKER; RARDIN, 1988). Por ser côncava, o ótimo local é ótimo global. Entretanto, por ser linear por partes não se pode garantir que seja diferenciável no ponto ótimo. Com isso, não se podem usar gradientes, que são substituídos por subgradientes e aproximam no sentido lagrangeano as soluções do problema relaxado à solução ótima (REEVES, 1995; ESPEJO; GALVÃO, 2002). A otimização do dual lagrangeano é obtida por meio do algoritmo de subgradientes, que é mostrado na Figura 2.5.

Passo 0: Inicialização.
Faça $u^1 \geq 0$, k ← 1, $v^0 \leftarrow -\infty$

Passo 1: Relaxação Lagrangeana
Resolver L_uP. Se $(Ax^k\text{-}b) \leq 0$ e $u(Ax^k\text{-}b) = 0$, pare, pois x^k resolve o problema primal P e $v(L_uP) = v(DL) = v(P)$.

Passo 2: Guardar a melhor solução dual
Se $v^{k-1} < v(L_uP) = cx^k + u(Ax^k\text{-}b)$, faça $v^k \leftarrow v(L_uP)$. Caso contrário, faça $v^k \leftarrow v^{k-1}$.

Passo 3: Calcular o passo do subgradiente
Ache um novo ponto
$u^{k+1} = u^k + \pi_k(Ax^k\text{-}b)/||Ax^k\text{-}b||$, sendo π_k o tamanho do passo que satisfaz as seguintes condições:

1) $\pi_k > 0$, para todo k

2) $\lim_{k\to\infty} \pi_k = 0$

3) $\sum_{k=1}^{\infty} \pi_k = +\infty$

Passo 4: Projeção
Projete o novo valor u^{k+1}, fazendo:
$u_j^{k+1} \leftarrow Max\{0, u_j^{k+1}\} \quad \forall j$
Faça k ← k + 1 e volte ao passo 1.

Figura 2.5 - Algoritmo de Subgradientes.

Fonte: Adaptado de Parker e Rardin (1988).

Dois detalhes são importantes sobre o algoritmo de subgradientes. Primeiro, como os passos do algoritmo não garantem melhorias em $v(L_uP)$, o valor da solução v^k é mantido no Passo 2. Segundo, o tamanho de π_k deve respeitar as condições descritas no Passo 3 para que haja convergência. Parker e Rardin (1998) apresentam um exemplo e provam que o algoritmo contém condições suficientes para garantir a convergência do método. Outros detalhes desse algoritmo podem ser encontrados em Narciso (1998).

2.3.2 Relaxação Lagrangeana com Divisão em *Clusters* (LagClus)

A relaxação lagrangeana com divisão em *clusters* surgiu do trabalho de Ribeiro (2007), que adaptou os resultados de Warrier et al. (2005), os quais desenvolveram um algoritmo *Branch-and-Price* para o problema de Máximo Conjunto Independente de Vértices com Pesos (PMCIVP). O PMCIVP pertence à classe de problemas que podem ser decompostos em subproblemas do mesmo tipo do original. A abordagem de Warrier et al. (2005) considera o particionamento do conjunto de vértices do grafo de conflitos para a obtenção de subgrafos induzidos de mais fácil solução. O problema original é então reformulado usando-se a decomposição Dantzig-Wolfe (BAZARAA et al., 1990) e esses subproblemas passam a gerar colunas para a decomposição, aproximando as soluções do problema original. A solução apresentada pelos autores mostra sucesso na abordagem, porém com tempos computacionais altos em alguns casos. Na tentativa de se obterem tempos mais adequados, surgiu a idéia de aplicar a relaxação lagrangeana a esse tipo de problema, desenvolvendo-se a LagClus (RIBEIRO, 2007).

A idéia da LagClus é a de trabalhar com problemas que podem ser decompostos em subproblemas com as mesmas características do problema original, obtendo-se assim problemas menores que podem ser resolvidos de forma exata em tempos computacionais aceitáveis com, por exemplo, algum *solver* comercial. Com isso, busca-se particionar um grafo que representa restrições do problema, separando-o em subgrafos com as mesmas características do original e relaxar no sentido lagrangeano as arestas (relacionadas a restrições) que estão entre os *clusters*. Bons resultados dessa técnica foram obtidos para o Problema de Estivagem de Unidades de Celulose (RIBEIRO; LORENA, 2008b), Problema de Carregamento de Paletes ao Produtor (RIBEIRO; LORENA, 2007), Problema de Rotulação Cartográfica de Pontos (RIBEIRO; LORENA, 2008a) e *Uncapacitated Facility Location Problem* (CORRÊA; LORENA, 2006). Em todos estes casos, o grafo utilizado foi o de conflitos.

O processo de aplicação da LagClus pode ser sintetizado nas seguintes ações:

- Montar o grafo (de conflitos ou de cobertura) $G = (V, E)$ do problema;

- Aplicar uma heurística de particionamento para dividir o grafo G em K *clusters* com aproximadamente a mesma cardinalidade, obtendo-se um K-particionamento. O problema passa a ser então representado por meio de sua função objetivo, sujeita às restrições divididas em dois conjuntos: um formado com as restrições que correspondem às arestas que conectam os *clusters* ou arestas de ligação; e outro com as demais restrições, que incluem as restrições correspondentes às arestas intra *clusters*;

- Relaxar, no sentido lagrangeano, as restrições relacionadas às arestas que conectam os *clusters*; e

- Decompor a relaxação lagrangeana resultante em K subproblemas e resolver o dual associado por meio de um algoritmo de subgradientes.

Os subproblemas gerados com este particionamento possuem, em menor escala, as mesmas características do problema original. Com isto, *solvers* comerciais e heurísticas eficientes podem resolvê-los de forma ótima. Entretanto, dependendo de seus tamanhos e de suas características, os subproblemas podem ser tão difíceis quanto o problema original. Neste caso, pode-se fazer o particionamento em um número maior de *clusters*, porém, deve-se avaliar a qualidade do limitante obtido. A LagClus pode ser mais ou menos forte dependendo do valor de K. Quanto menor for K, mais forte é a relaxação (RIBEIRO, 2007).

Resumindo, a principal idéia da LagClus é a divisão de um grafo em *clusters* e a relaxação no sentido lagrangeano das arestas de conexão entre esses *clusters*.

2.4 Decomposição Dantzig-Wolfe

Considere uma empresa com vários departamentos. Cada um tem o seu planejamento a fazer, bem como restrições em suas decisões. Além disto, algumas restrições se aplicam a todos os departamentos, como por exemplo, o orçamento global. Baseando-se nas restrições comuns, a cada departamento é solicitada a formulação de uma proposta que indique o seu plano ótimo. Um planejador central reúne essas propostas e as combina com propostas já existentes para montar um plano global para a empresa. Esse novo plano determina novos valores para as restrições comuns. O processo é repetido para verificar se algum departamento pode melhorar o plano global. Neste caso, pode-se considerar que o problema tem a seguinte estrutura:

a) Vários grupos lógicos de variáveis.

b) Muitas restrições afetam apenas um grupo de variáveis.

c) Algumas restrições acoplam os grupos de variáveis.

Este tipo de estrutura é forte candidata a ser tratada pela idéia de "dividir para conquistar", sendo uma das técnicas a decomposição Dantzig-Wolfe (DANTZIG; WOLFE, 1960). Uma maneira formal de descrevê-la pode ser encontrada em Wolsey (1998) e Ribeiro (2007), que será resumida a seguir.

Considere um problema $(P) : Max\{cx : x \in X\}$ com uma região factível X que pode ser descrita como a intersecção de dois ou mais conjuntos com a estrutura $X = \cap_{k=1}^{K} \mathbf{X}^k$, para algum $K > 1$. Considere (P) como definido a seguir, em que x é o vetor de variáveis de decisão, c é o vetor de custos, b e d são vetores de disponibilidades de recursos e A e D são matrizes de coeficientes:

$$v(P) = Max \sum_{k=1}^{K} c^k x^k \tag{2.26}$$

Sujeito a

$$A^1x^1 + A^2x^2 + ... + A^Kx^K = b \tag{2.27}$$

$$\begin{array}{rrrcl} D^1x^1 & & & \leq & d_1 \\ & ... & & \leq & . \\ & & ... & \leq & . \\ & & D^Kx^K & \leq & d_K \end{array} \tag{2.28}$$

$$x^1 \in Z_+^{n_1}, \quad ... \quad ... \quad , x^K \in Z_+^{n_K} \tag{2.29}$$

de maneira que os conjuntos $X^k = \left\{x^k \in Z_+^{n_k} : D^kx^k \leq d_k\right\}$ sejam independentes para $k = 1, ..., K$, e que somente as restrições $\sum_{k=1}^{K} A^kx^k = b$ acoplem os diferentes conjuntos de variáveis. Considere m a cardinalidade do vetor b.

A relaxação lagrangeana discutida anteriormente permite que se tire vantagens da estrutura definida em 2.26-2.29, pois ao relaxar 2.27, o

problema relaxado L_uP pode ser decomposto em K distintos subproblemas $\left\{v\left(L_uP\right) = \sum_{k=1}^{K} Max\left(\left(c^k - uA^k\right)x^k : x^k \in X^k\right) + ub\right\}$, uma vez que as restrições restantes são independentes entre si.

Este problema apresenta as características descritas no exemplo da empresa e seus departamentos, ou seja, formados por blocos de restrições independentes (departamentos, com suas decisões e restrições) e por um único bloco de acoplamentos (restrições 2.27, que correspondem às restrições globais da empresa).

Considerando que cada conjunto $X^k = \left\{x^k \in Z_+^{n_k} : D^k x^k \leq d_k\right\}$ possui um finito conjunto T_k de soluções (pontos extremos) $\left\{x^{k,t}\right\}_{t=1}^{T_k}$, cada ponto do conjunto X^k pode ser escrito como uma combinação linear convexa de seus pontos extremos:

$$\left\{x^k \in Z_+^{n_k} : x^k = \sum_{t=1}^{T_k} \lambda_{k,t} x^{k,t}, \sum_{t=1}^{T_k} \lambda_{k,t} = 1, \lambda_{k,t} \in \{0,1\}\, \forall t = 1, ..., T_k\right\} \tag{2.30}$$

Assim, o problema P pode ser definido como:

$$v(P) = Max \sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(c^k x^{k,t}\right)\lambda_{k,t} \tag{2.31}$$

Sujeito a

$$\sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(A^k x^{k,t}\right)\lambda_{k,t} = b \tag{2.32}$$

$$\sum_{t=1}^{T_k} \lambda_{k,t} = 1 \quad \forall k = 1, ..., K \tag{2.33}$$

$$\lambda_{k,t} \in \{0,1\} \quad \forall t = 1, ..., T_k,\ k = 1, ..., K \tag{2.34}$$

O problema de encontrar soluções viáveis para X^k é chamado de subproblema e o problema descrito em 2.31-2.34 é denominado Problema Mestre (PM). Estes dois problemas definem a decomposição Dantzig-Wolfe. No Problema Mestre, as restrições definidas em 2.32 são conhecidas como restrições de acoplamento e as definidas em 2.33, como restrições de convexidade.

Considerando a relaxação de programação linear do $PM(PM_{PL})$, tem-se:

$$v(PM_{PL}) = Max \sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(c^k x^{k,t}\right) \lambda_{k,t} \tag{2.35}$$

Sujeito a

$$\sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(A^k x^{k,t}\right) \lambda_{k,t} = b \tag{2.36}$$

$$\sum_{t=1}^{T_k} \lambda_{k,t} = 1 \quad \forall k = 1, ..., K \tag{2.37}$$

$$\lambda_{k,t} \geq 0 \quad \forall t = 1, ..., T_k, \ k = 1, ..., K \tag{2.38}$$

Seja $\begin{pmatrix} c^k x \\ A^k x \end{pmatrix}$ uma coluna para cada $x \in X^k$, $\quad \{\alpha_i\}_{i=1}^{m}$ as variáveis duais associadas às m restrições de acoplamento e $\{\beta_k\}_{k=1}^{K}$ as variáveis duais associadas às K restrições de convexidade.

Desta forma, e infelizmente, existem mais variáveis no PM que na formulação original, pois é criada uma variável para cada ponto extremo do politopo do subproblema $X^k = \left\{x^k \in Z_+^{n_k} : D^k x^k \leq d_k\right\}$, e não é prático armazenar os coeficientes para todos os pontos extremos de um problema real (Martin, 1999). Isto motiva a idéia de se trabalhar com um subconjunto $X^{k,l}$ que representa os l pontos extremos de X^k $\left(X^{k,l} \subseteq X^k\right)$. O Problema Mestre quando considera apenas um subconjunto de pontos extremos $X^{k,l}$ para cada k, passa a ser denominado Problema Mestre Restrito (PMR).

Assim o Algoritmo de Geração de Colunas, mostrado a seguir, parte da idéia inicial de se trabalhar com um subconjunto de pontos extremos e, em seguida, gerar novos pontos extremos de maneira sistemática. Supondo que exista um subconjunto de colunas (pelo menos uma para cada k), a relaxação de programação linear para o PMR pode ser obtida:

$$\tilde{v}(PMR_{PL}) = Max \ \tilde{c}\tilde{\lambda} \tag{2.39}$$

Sujeito a

$$\tilde{A}\tilde{\lambda} = \tilde{b} \tag{2.40}$$

$$\tilde{\lambda} \geq 0 \tag{2.41}$$

Em que $\tilde{b} = \begin{pmatrix} b \\ \mathbf{1} \end{pmatrix}$, $\tilde{c}$ e $\tilde{\lambda}$ são os correspondentes custos e variáveis e $\tilde{A}$ é uma matriz gerada conforme o número de colunas disponíveis, descrita da seguinte forma:

$$\begin{bmatrix} A^1x^{1,1} & \dots & A^1x^{1,T_1} & A^2x^{2,1} & \dots & A^2x^{2,T_2} & \dots & A^Kx^{K,1} & \dots & A^Kx^{K,T_K} \\ 1 & \dots & 1 & & & & & & & \\ & & & 1 & \dots & 1 & & & & \\ & & & & & & \dots & & & \\ & & & & & & & 1 & \dots & 1 \end{bmatrix}$$

Ao resolver o PMR_{PL}, obtém-se uma solução ótima $\tilde{\lambda}^*$ do problema e uma solução ótima dual $(\alpha, \beta) \in \Re^m \mathrm{x}\, \Re^K$. Qualquer solução viável do PMR_{PL} é viável para PM_{PL}. Em particular, se $\tilde{\lambda}^*$ é uma solução viável de PM_{PL}, então $\tilde{v}(PMR_{PL}) = \tilde{c}\tilde{\lambda}^* = \sum_{i=1}^{m} \alpha_i b_i + \sum_{k=1}^{K} \beta_k \leq v\,(PM_{PL})$.

É necessário verificar se (α, β) é um dual viável para o PM_{PL}. Isso implica verificar para cada coluna, para cada k e para $x \in X^k$ se o custo reduzido $c^k x - \alpha A^k x - \beta_k \leq 0$. Entretanto, em vez de tratar cada ponto separadamente, tratam-se todos os pontos em X^k implicitamente ao resolver um subproblema de otimização:

$$v\,(\zeta_k) = Max\left\{\left(c^k - \alpha A^k\right) x - \beta_k : x \in X^k\right\} \quad \forall k = 1, ..., K \tag{2.42}$$

Se $v(\zeta_k) > 0$ para algum k, a solução correspondente à solução ótima $\tilde{x}^k$ do subproblema tem um custo reduzido positivo, então a coluna $\begin{pmatrix} c^k\tilde{x}^k \\ A^k\tilde{x}^k \end{pmatrix}$ deve ser inserida no PMR_{PL} e o novo PMR deve ser resolvido.

Do subproblema, tem-se que $v\,(\zeta_k) \geq \left(c^k - \alpha A^k\right) x - \beta_k$ para todo $x \in X^k$. Isto implica que $\left(c^k - \alpha A^k\right) x - \beta_k - v(\zeta^k) \leq 0$ para todo $x \in X^k$. Assim, sendo $\zeta = (v\,(\zeta_1), ..., v\,(\zeta_K))$, tem-se que $(\alpha, \beta + \zeta)$ é um dual viável para o PM_{PL}. Portanto,

$$v\left(PM_{PL}\right) \leq \alpha b + \sum_{k=1}^{K} \left(\beta_k + v\left(\zeta_k\right)\right) \tag{2.43}$$

As definições acima conduzem a um algoritmo para PM_{PL} que termina quando $v\left(\zeta_k\right) = 0$, para todo $k = 1, ..., K$, que é denominado Algoritmo de Geração de Colunas. A Figura 2.6 apresenta um esquema básico desse algoritmo.

Passo 0 : Inicialização
Determine um conjunto inicial de colunas para o PMR.
Passo 1: Resolução do PMR_{PL}.
Resolva o PMR_{PL}.
Obtenha os valores duais (α, β).
Passo 2: Atualização e resolução dos subproblemas geradores de colunas.
Com os valores duais (α, β) do PMR_{PL}, atualize e resolva os subproblemas.
Se $\zeta_k = 0$, para $k = 1,\ldots,K$, vá para o Passo 4.
Se $\zeta_k > 0$, para algum k, vá ao Passo 3.
Passo 3: Adicione colunas no PMR_{PL}.
Adicione as novas colunas com custo positivo no PMR_{PL}.
Volte ao Passo 1.
Passo 4: Pare, pois a solução é ótima.

Figura 2.6 - Algoritmo básico de geração de colunas.

Fonte: Adaptada de Ribeiro (2007).

2.5 Algoritmo Genético Construtivo

O algoritmo genético foi concebido por Holland, em 1960, e aperfeiçoado nas décadas de 1960 e 1970, por ele próprio e por seus colegas da Universidade de Michigan. Vários refinamentos do método surgiram nas décadas seguintes. Alguns deles e informações básicas podem ser vistas em Goldberg (1989), Lacerda e Carvalho (1999).

O Algoritmo Genético Construtivo (AGC) (FURTADO, 1998) trabalha com uma população inicial composta por estruturas e esquemas. As estruturas referem-se a qualquer cadeia contendo ou não o símbolo #, chamado de curinga e representa um ponto ainda não definido para o problema; os esquemas fazem referência explícita a uma estrutura que contém o símbolo #, isto é, uma população de soluções candidatas parciais ou incompletas, que servirão de base para a construção de uma população com soluções melhores

completas, ao longo do processo evolutivo.

O AGC possui os operadores tradicionais de seleção, cruzamento e mutação, e difere dos algoritmos genéticos convencionais na forma de avaliar os esquemas (avaliação-*fg*), na possibilidade de usar heurísticas para definir a função de avaliação da aptidão dos indivíduos e no tratamento de uma população de tamanho variável (OLIVEIRA; LORENA, 2004; OLIVEIRA, 2004).

A avaliação-*fg* trata de uma dupla avaliação de cada indivíduo $S_k \in P_e$, que é a população no instante de evolução e. Na maximização, o valor de f reflete a função objetivo, $f : P_e \rightarrow R^+$, e o valor de g é calculado a partir de uma heurística, $g : P_e \rightarrow R^+$, tal que $g(S_k) \geq f(S_k)$, para todo $S_k \in P_e$. A primeira, função f, avalia a qualidade do indivíduo e a segunda, função g, aplica uma heurística para avaliar a vizinhança do indivíduo, atribuindo a melhor solução encontrada ao valor de g. O indivíduo original não estará bem adaptado se a heurística encontrar uma solução melhor. Caso contrário, o indivíduo é o melhor dentro da vizinhança estabelecida pela heurística e deve participar o máximo possível do processo evolutivo. Essa dupla avaliação é aplicada da mesma forma para qualquer tipo de indivíduo S_k, assim considerados os esquemas e estruturas. Deve ser definido também um limite superior comum $G_{Max} > \underset{S_k \in P_e}{Max}\ g(S_k)$.

O AGC trabalha com uma população de tamanho variável, com a população inicial gerada aleatoriamente. A cada geração do processo evolutivo, novos indivíduos são criados, um para cada cruzamento entre dois indivíduos existentes, privilegiando os mais bem adaptados. Esses novos indivíduos podem sofrer mutação, dentro de uma certa probabilidade definida *a priori*.

Um problema a ser modelado em AGC deve ser tratado como um problema de otimização bi-objetivo (PBO):

$$\begin{cases} \min\{g(S_k) - f(S_k)\} \\ \max\{f(S_k)\} \\ Sujeito\ a\ g(S_k) \geq f(S_k) \end{cases} \tag{2.44}$$

Para qualquer problema de otimização, o PBO é formulado da mesma forma: maximizar a função objetivo e minimizar o intervalo (g-f). O processo evolutivo considera um limiar de rejeição adaptativo que contempla ambos os objetivos do PBO, que é calculado a partir de um *ranking* δ atribuído a cada indivíduo da população, dado pela equação 2.45, que é composta de:

a) Um componente referente à adaptação do indivíduo: (g-f). Quanto menor essa diferença, mais adaptado está o indivíduo.

b) Um componente que privilegia a maximização da função g, definido pela distância entre o valor de g e o limitante superior G_{max}.

c) Uma constante $d \in [0, 1]$, que tem o papel de equilibrar os componentes da equação.

$$\delta = \frac{d.G_{\max} - [g(S_k) - f(S_k)]}{d.\,[G_{\max} - g(S_k)]} \tag{2.45}$$

A população inicial, denominada P_0, é formada apenas por esquemas. Considerando como exemplo o Problema das p-Medianas, para cada indivíduo, uma porcentagem das posições recebe aleatoriamente o valor *2* e exatamente p posições recebem o valor *1*, definindo, assim, as p facilidades a serem abertas. As posições restantes recebem o valor #. A Figura 2.7 mostra um possível indivíduo da população inicial para um problema com quatro centros e trinta nós.

2 # 1 # # # # 2 # # 1 # 2 # # 2 # 2 # # 1 # # # 1 # 2

Figura 2.7 - Possível indivíduo para um problema com quatro centros e trinta nós.

Os indivíduos são ordenados por valores decrescentes de δ. O tamanho da população é controlado dinamicamente pelo limiar de rejeição e, que está relacionado com o instante de evolução e serve para eliminar os indivíduos mal adaptados ($\delta\,(S_k) \leq e$). O valor de e é calculado pela equação 2.46, em que δ_1 define o melhor valor de *ranking*; $\delta_{|P|}$, o pior valor de *ranking*; ε é uma constante que controla a velocidade de esvaziamento da população; $|P|$ é o tamanho da população no instante e; RG representa o número de gerações restantes no processo evolutivo e l é um valor que garante um passo mínimo nesse processo.

$$e_t = e_{t-1} + \varepsilon.|P|.\frac{(\delta_1 - \delta_{|P|})}{RG} + l \tag{2.46}$$

Considerando-se que os bons esquemas precisam ser preservados para serem recombinados, e é iniciado com o valor 0 (zero) e é lentamente incrementado, de geração em geração.

Desde que são criados, os esquemas e estruturas recebem os seus correspondentes valores

de *ranking* (δ), que são comparados com o parâmetro e. Todo indivíduo com delta (S_k) $\leq e$ é considerado mal adaptado e é eliminado da população. Conseqüentemente, os indivíduos com maiores δ são os melhores em relação ao PBO, sobrevivem por mais gerações e se reproduzem mais. Com isso, a população no tempo de evolução e (P_e) possui tamanho dinâmico de acordo com o valor de e, conforme exemplo na Figura 2.8, podendo ser esvaziada durante o processo. Sempre o melhor indivíduo de cada geração é guardado e, ao final, o processo evolutivo retorna a melhor solução encontrada.

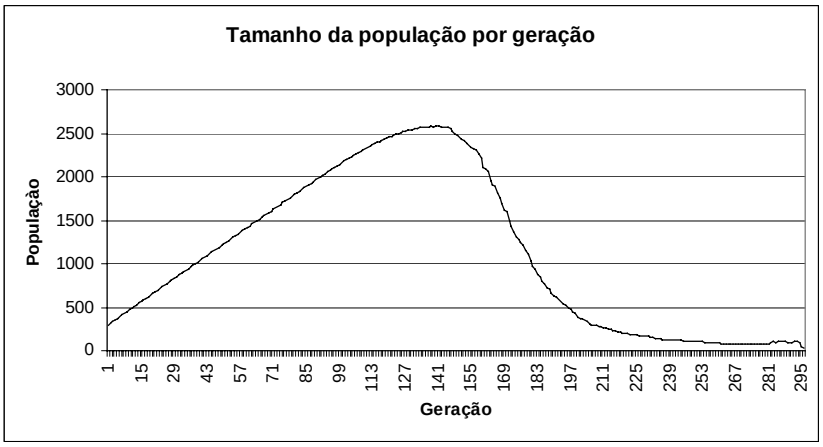


Figura 2.8 - Tamanho da população a cada geração.

Fonte: Adaptada de Furtado (1998).

O operador de seleção base-guia (Furtado, 1998) escolhe duas estruturas para o cruzamento. A primeira, denominada base (S_{base}), é obtida dos melhores indivíduos da população. A segunda, denominada guia (S_{guia}), é selecionada da população total. O operador de cruzamento compara essas duas estruturas em posições correspondentes, gerando uma nova estrutura filha (S_{nova}).

O algoritmo para o AGC pode ser resumido no pseudocódigo mostrado na Figura 2.9.

Defina G_{max}, d, ε, l;
e = 0;
Defina a probabilidade de mutação;
Defina a população inicial (P_e);
Para todo $S_k \in P_e$ **faça**
    Calcule $f(S_k)$, $g(S_k)$, $\delta(S_k)$; // O cálculo de $\delta(S_k)$ é feito segundo a fórmula (2.45)
Fim-Para;
Enquanto (condição de parada não for satisfeita) **faça**
    Enquanto (não atingir a quantidade de cruzamentos prevista) **faça**
        Seleção (S_{base}, S_{guia});
        S_{nova} = Cruzamento (S_{base}, S_{guia});
        Viabilize S_{nova};
        Se satisfeita a probabilidade de mutação
            Então
                S_{nova} = Mutação(S_{nova});
        Fim-Se;
        Calcule $f(S_{nova})$, $g(S_{nova})$, $\delta(S_{nova})$;
        Atualize (P_e, S_{nova});
    Fim-Enquanto;
    Atualize o valor de e usando a fórmula (2.46)
    Para todo ($S_k \in P_e$) e ($\delta(S_k) < \alpha$) **faça**
        Elimine (S_k, P_e)
    Fim-Para;
Fim-Enquanto;

Figura 2.9 - Pseudocódigo para o AGC.

2.6 *Clustering Search* (CS)

A metaheurística *Clustering Search* (CS), proposta por Oliveira e Lorena (2004), consiste num processo de agrupamentos de soluções para detectar regiões supostamente promissoras no espaço de busca, isto é, localizar regiões representadas por agrupamentos de soluções com características similares. Deste modo, uma região pode ser vista como um subespaço de busca definido por uma relação de vizinhança. O termo *detecção de regiões promissoras* está associado a mecanismos capazes de identificar o potencial positivo de certos subespaços de busca, gerando a possibilidade de interferência nas estratégias de busca associadas com cada um deles (OLIVEIRA, 2004).

As regiões formadas por esses grupos de soluções devem ser exploradas por meio de heurísticas de busca local específicas, tão logo sejam consideradas promissoras. Busca-se uma melhoria no processo de convergência associada a uma diminuição no esforço computacional, em virtude do emprego mais racional dos métodos de busca local.

Um agrupamento é definido pela tripla $G = \{C; r; E\}$, em que C, r e E são, respectivamente, o centróide e o raio de uma região promissora, e uma estratégia de busca. O centróide C é uma solução que representa o agrupamento, identificando a sua localização em uma região dentro do espaço de busca. Inicialmente, os centróides são obtidos aleatoriamente e, progressivamente, tendem a deslocar-se para pontos realmente promissores no espaço de busca. O raio r estabelece a distância máxima, a partir do centróide da região, até a qual uma solução pode ser associada ao agrupamento. A estratégia de busca E é uma sistemática de intensificação de busca, na qual soluções de um agrupamento interagem entre si, ao longo do processo de encontrar similaridades, gerando novas soluções.

O CS consiste em quatro componentes conceitualmente independentes com diferentes atribuições:

a) Uma metaheurística (MH).

b) Um agrupador iterativo (AI).

c) Um analisador de agrupamentos (AA).

d) Um algoritmo de otimização local (AO).

O componente *MH* trabalha como um gerador de soluções de tempo integral. O algoritmo é executado independentemente dos componentes restantes e precisa ser capaz de gerar soluções de forma contínua para a busca por similaridades. Simultaneamente, agrupamentos são mantidos para representar estas soluções. Este processo trabalha como um laço infinito, no qual soluções são geradas ao longo das iterações.

O componente *AI* objetiva reunir soluções similares dentro de um agrupamento, mantendo uma solução no seu centróide que seja representativa para as demais soluções. Um número máximo de agrupamentos é definido *a priori*. Uma métrica de distância também precisa ser definida inicialmente, que representa uma medida de similaridade. Toda solução gerada pelo *MH* é inserida no agrupamento que lhe for mais similar, causando uma perturbação no seu centróide. Tal perturbação é chamada de assimilação e consiste basicamente em atualizar o centróide, considerando as suas informações e as da nova solução gerada.

O componente *AA* provê uma análise de cada agrupamento em intervalos regulares, indicando se é ou não, promissor. Esta indicação é dada pelo nível de atividade dentro do agrupamento, isto é, pela quantidade de soluções a ele designadas. Sempre que essa quantidade atingir certo valor χ_i, o agrupamento i precisa ser mais bem investigado para acelerar o seu processo de convergência.

Por fim, o componente AO é um módulo de busca local que provê a exploração de uma suposta região promissora, isto é, uma estratégia de busca E associada a um agrupamento. Este processo acontece após AA ter descoberto uma região promissora. Com isto, uma busca local é aplicada ao seu centróide.

2.7 Considerações Finais

Este capítulo apresentou a fundamentação teórica necessária à discussão dos tópicos a serem tratados nos próximos capítulos deste trabalho. Foram abordados os problemas de localização de facilidades, grafos, relaxações, método de geração de colunas, a metaheurística Algoritmo Genético Construtivo e *Clustering Search*.

No próximo Capítulo será discutido o Problema de Localização de Facilidades Não-capacitado e apresentadas três soluções para esse problema: relaxação lagrangeana, LagClus e Algoritmo de Geração de Colunas, sendo as duas últimas implementadas a partir da modelagem do problema feita por grafo de conflitos.

3 O PROBLEMA DE LOCALIZAÇÃO DE FACILIDADES NÃO-CAPACITADO

Neste Capítulo, apresentam-se três soluções para problema de localização de facilidades não-capacitado. Duas são baseadas na modelagem em grafo de conflitos: uma utilizando a LagClus e outra, o Algoritmo de Geração de Colunas. A terceira é baseada na relaxação lagrangeana tradicional. Apresentam-se, ainda, os resultados computacionais desses métodos aplicados a instâncias com grande *gap* de dualidade e de difícil solução para a relaxação de programação linear (KOCHETOV; IVANENKO, 2003) e de algumas obtidas de *OR-Library* (BEASLEY, 1990).

3.1 Formulações para o Problema de Localização de Facilidades Não-Capacitado

O UFLP foi apresentado no Capítulo 2 com a formulação mostrada a seguir:

$$v(UFLP) = Min \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + \sum_{j \in J} f_j y_j \tag{3.1}$$

Sujeito a

$$\sum_{j \in J} x_{ij} = 1 \quad \forall i \in I \tag{3.2}$$

$$x_{ij} \leq y_j \quad \forall i \in I,\ \forall j \in J \tag{3.3}$$

$$x_{ij} \in \{0,1\};\ y_j \in \{0,1\} \quad \forall i \in I,\ \forall j \in J \tag{3.4}$$

O modelo tradicional acima proposto pode ser alterado para comportar restrições de adjacência ou de conflitos. Considerando então o complemento das variáveis de localização, ou seja, $\bar{y}_j = 1 - y_j$, conforme indicado por Cornuéjols e Thizy (1982), a formulação do UFLP definida em ((3.1)-(3.4)) pode ser re-escrita como mostrado a seguir. Os conflitos aparecem nas restrições (3.6) e nas restrições definidas em (3.7), que são as restrições (3.3) modificadas, em que apenas a variável de alocação ou a de localização pode receber o valor um.

$$v(\overline{UFLP}) = Min \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} - \sum_{j \in J} f_j \bar{y}_j + \sum_{j \in J} f_j \tag{3.5}$$

Sujeito a

$$\sum_{j \in J} x_{ij} = 1 \quad \forall i \in I \tag{3.6}$$

$$x_{ij} + \bar{y}_j \leq 1 \quad \forall i \in I,\ \forall j \in J \tag{3.7}$$

$$x_{ij}, \bar{y}_j \in \{0, 1\} \quad \forall i \in I,\ \forall j \in J \tag{3.8}$$

Como exemplo, considere um problema com $m = |I| = 4$ *e* $n = |J| = 4$. Com isso, o problema $\overline{UFLP}$ pode ser assim escrito:

$v\left(\overline{UFLP}\right) = Min\{c_{11}x_{11}+c_{12}x_{12}+c_{13}x_{13}+c_{14}x_{14}+c_{21}x_{21}+c_{22}x_{22}+c_{23}x_{23}+c_{24}x_{24}+$
$c_{31}x_{31}+c_{32}x_{32}+c_{33}x_{33}+c_{34}x_{34}+c_{41}x_{41}+c_{42}x_{42}+c_{43}x_{43}+c_{44}x_{44}$
$-f_1\bar{y}_1-f_2\bar{y}_2-f_3\bar{y}_3-f_4\bar{y}_4+f_1+f_2+f_3+f_4\}$

Sujeito a

$x_{11}+x_{12}+x_{13}+x_{14}= 1$
$x_{21}+x_{22}+x_{23}+x_{24}= 1$
$x_{31}+x_{32}+x_{33}+x_{34}= 1$
$x_{41}+x_{42}+x_{43}+x_{44}= 1$
$x_{11}+\bar{y}_1<= 1$
$x_{12}+\bar{y}_2<= 1$
$x_{13}+\bar{y}_3<= 1$
$x_{14}+\bar{y}_4<= 1$
$x_{21}+\bar{y}_1<= 1$
$x_{22}+\bar{y}_2<= 1$
$x_{23}+\bar{y}_3<= 1$
$x_{24}+\bar{y}_4<= 1$
$x_{31}+\bar{y}_1<= 1$
$x_{32}+\bar{y}_2<= 1$
$x_{33}+\bar{y}_3<= 1$
$x_{34}+\bar{y}_4<= 1$

$x_{41}+\bar{y}_1<=1$
$x_{42}+\bar{y}_2<=1$
$x_{43}+\bar{y}_3<=1$
$x_{44}+\bar{y}_4<=1$
$x_{11}, x_{12}, x_{13}, x_{14}, x_{21}, x_{22}, x_{23}, x_{24}, x_{31}, x_{32}, x_{33}, x_{34}, x_{41}, x_{42}, x_{43}, x_{44}, \bar{y}_1, \bar{y}_2, \bar{y}_3, \bar{y}_4 \in \{0,1\}$

A Figura 3.1 mostra o grafo de conflitos referente ao exemplo.

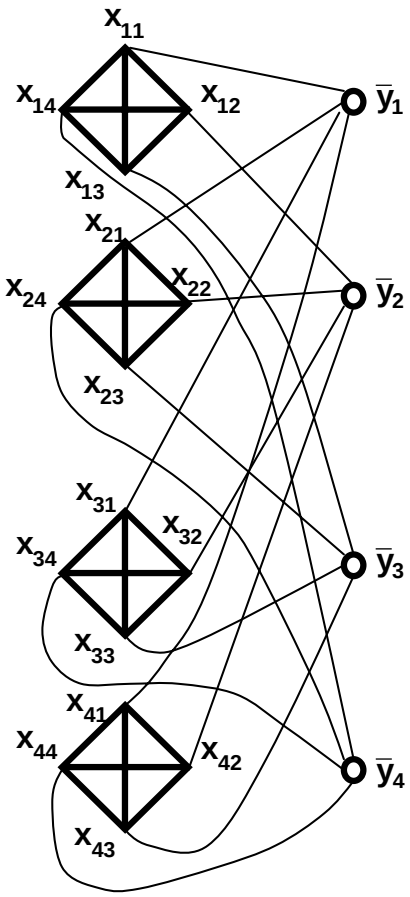


Figura 3.1 - Grafo de conflitos referente ao problema $\overline{UFLP}$ com $m = n = 4$.

3.2 Aplicação da LagClus ao $\overline{UFLP}$

Depois de definido o grafo de conflitos, deve-se particioná-lo em K subgrafos, de forma que as restrições de adjacências sejam divididas em intra e entre *clusters*. Antes desse particionamento, é feito o colapso das cliques (restrições (3.6)) para garantir que todos os seus vértices permaneçam no mesmo *cluster*. Isto evita o particionamento das arestas das cliques.

Particionando o grafo de conflitos G = (V,E) em K partes $V_1, V_2, \ldots, V_K$ define-se subgrafos $G_k = G[V_k]$ com o conjunto de arestas $E_k = E(G_k)$. As arestas de G que conectam os subgrafos, isto é, que suas extremidades estão em diferentes conjuntos da partição, compreendem o conjunto $\hat{E} = E \setminus \cup_{k=1}^{K} E_k$.

Aplicando-se uma heurística de particionamento ao grafo de conflitos de $\overline{UFLP}$ obtém-se a seguinte representação matricial, em que o termo $\sum_{j \in J} f_j$, por conter apenas valores constantes, será inserido no cálculo final do limitante lagrangeano:

$$v\left(\overline{UFLP}\right) = Min \sum_{k=1}^{K} \mathbf{c}^k \mathbf{x}^k - \mathbf{f}^k \bar{\mathbf{y}}^k \tag{3.9}$$

Sujeito a

$$\begin{bmatrix} A_1 & A_2 & ... & A_K \\ B_1 & 0 & 0 & 0 \\ 0 & B_2 & 0 & 0 \\ ... & ... & ... & ... \\ 0 & 0 & 0 & B_K \end{bmatrix} \begin{pmatrix} \begin{bmatrix} \mathbf{x} \\ \bar{\mathbf{y}} \end{bmatrix}^1 \\ \begin{bmatrix} \mathbf{x} \\ \bar{\mathbf{y}} \end{bmatrix}^2 \\ \vdots \\ \begin{bmatrix} \mathbf{x} \\ \bar{\mathbf{y}} \end{bmatrix}^K \end{pmatrix} \approx \mathbf{1}, \quad \mathbf{x}^k, \bar{\mathbf{y}}^k \in B^{|V_k|} \quad \forall k \in \{1, .., K\} \tag{3.10}$$

Em que:

- A_k é a matriz de dimensões $|\hat{E}| \times |V_k|$ de coeficientes das inequações associadas com as arestas de conflitos pertencentes a $\hat{E}$.
- B_k é a matriz de dimensões $E - |\hat{E}| \times |V_k|$ de coeficientes das equações e inequações associadas com as arestas de E_k.
- $\approx$ representa os operadores relacionais $=$ ou $\leq$ dependendo das restrições associadas (equação ou inequação).

O subproblema k, $k \in 1, ..., K$, para o $\overline{UFLP}$ tem a forma:

$$v\left(\overline{UFLP}\right)_k^{LC} = Min\left\{\left(\mathbf{c}^k + A_k^T \mu\right)\mathbf{x}^k - \left(\mathbf{f}^k - A_k^T \mu\right)\bar{\mathbf{y}}^k : \mathbf{x}^k, \bar{\mathbf{y}}^k \in O_k\right\} \tag{3.11}$$

Em que $\mu \in R_+^Q$ são os multiplicadores lagrangeanos associados com as linhas da matriz A_k, Q é a quantidade de restrições relaxadas e O_k são as restrições associadas ao subproblema k (*cluster* k). As restrições que definem A_k são usadas para obter os subgradientes aplicados ao algoritmo descrito no Capítulo 2.

Aplicando-se a relaxação lagrangeana sobre esse conjunto de restrições e resolvendo-se os subproblemas $v\left(\overline{UFLP}\right)_k^{LC}$, o valor do limitante da LagClus ($LC_\mu\overline{UFLP}$) será dado por:

$$v\left(LC_\mu\overline{UFLP}\right) = \sum_{k=1}^{K} v\left(\overline{UFLP}\right)_k^{LC} - \sum_{q=1}^{Q} \mu_q + \sum_{j\in J} f_j \tag{3.12}$$

Continuando o exemplo iniciado na Seção 3.1, supondo-se a divisão em dois *clusters*, com os vértices 1 e 2 no *cluster* 1 e os vértices 3 e 4 no *cluster* 2, obtém-se a Figura 3.2 (a). As restrições relacionadas às arestas dos *clusters* 1 e 2 são mostradas na Tabela 3.1. Depois de removidas as arestas entre *clusters* (arestas em negrito), os subproblemas resultantes do particionamento são mostrados na Figura 3.2 (b).

Tabela 3.1 - Restrições relacionadas às arestas intra e entre *clusters*.

Arestas intra *clusters*		Arestas entre *clusters*	
Cluster 1	*Cluster* 2		
$x_{11} + \overline{y}_1 <= 1$	$x_{33} + \overline{y}_3 <= 1$	$x_{13} + \overline{y}_3 <= 1$	$x_{31} + \overline{y}_1 <= 1$
$x_{12} + \overline{y}_2 <= 1$	$x_{34} + \overline{y}_4 <= 1$	$x_{14} + \overline{y}_4 <= 1$	$x_{32} + \overline{y}_2 <= 1$
$x_{21} + \overline{y}_1 <= 1$	$x_{43} + \overline{y}_3 <= 1$	$x_{23} + \overline{y}_3 <= 1$	$x_{41} + \overline{y}_1 <= 1$
$x_{22} + \overline{y}_2 <= 1$	$x_{44} + \overline{y}_4 <= 1$	$x_{24} + \overline{y}_4 <= 1$	$x_{42} + \overline{y}_2 <= 1$

Após o particionamento, deve-se relaxar no sentido lagrangeano as restrições de adjacências entre *clusters* ($Q = 8$). A Tabela 3.2 mostra as arestas relaxadas e os correspondentes subgradientes.

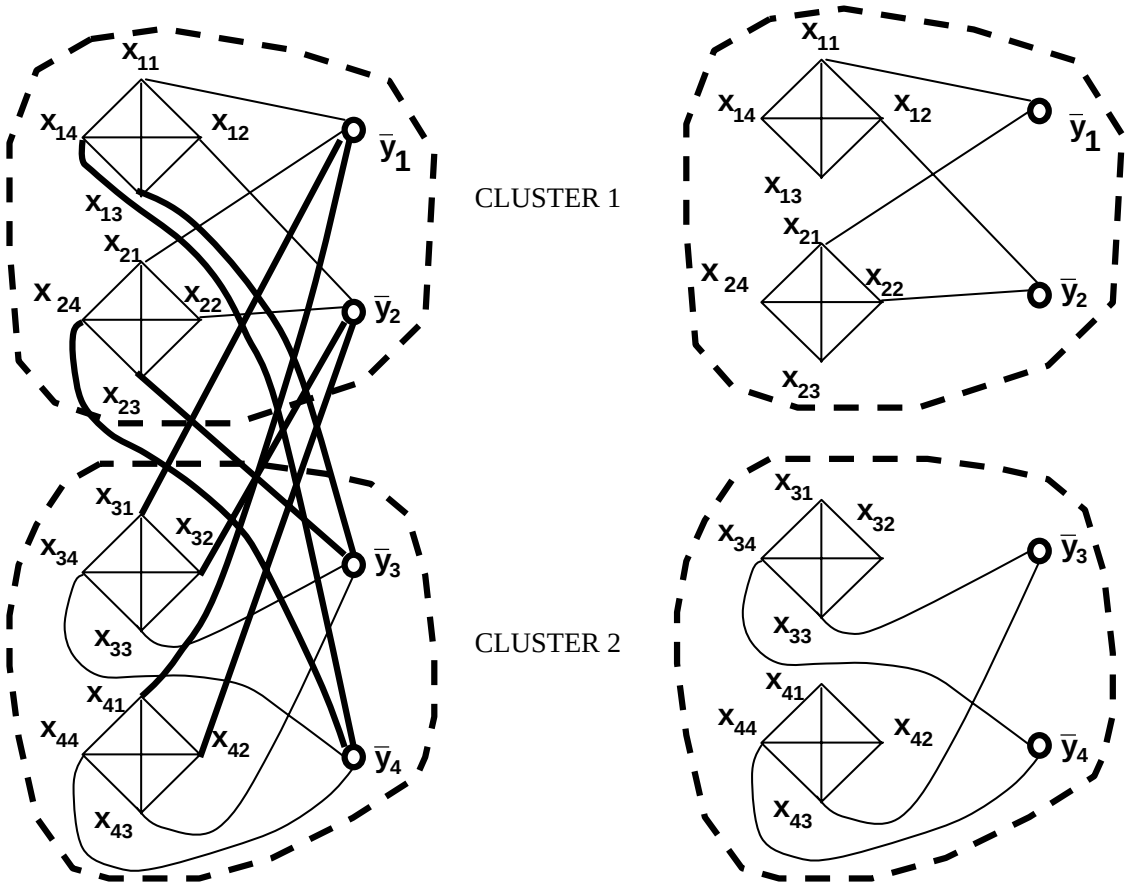


Figura 3.2 - (a) Particionamento em dois *clusters*, (b) *Clusters* finais.

Tabela 3.2 - Subgradientes

Restrições relaxadas	Subgradientes
$x_{13} + \overline{y}_3 <= 1$	$x_{13} + \overline{y}_3 - 1$
$x_{14} + \overline{y}_4 <= 1$	$x_{14} + \overline{y}_4 - 1$
$x_{23} + \overline{y}_3 <= 1$	$x_{23} + \overline{y}_3 - 1$
$x_{24} + \overline{y}_4 <= 1$	$x_{24} + \overline{y}_4 - 1$
$x_{31} + \overline{y}_1 <= 1$	$x_{31} + \overline{y}_1 - 1$
$x_{32} + \overline{y}_2 <= 1$	$x_{32} + \overline{y}_2 - 1$
$x_{41} + \overline{y}_1 <= 1$	$x_{41} + \overline{y}_1 - 1$
$x_{42} + \overline{y}_2 <= 1$	$x_{42} + \overline{y}_2 - 1$

Com os subgradientes incorporados à formulação do problema, tem-se:

$$
\begin{aligned}
v\left(\overline{UFLP}\right)^{LC} = Min\ \{ & c_{11}x_{11}+c_{12}x_{12}+(c_{13}+\mu_1)x_{13}+(c_{14}+\mu_2)x_{14}+ \\
& c_{21}x_{21}+c_{22}x_{22}+(c_{23}+\mu_3)x_{23}+(c_{24}+\mu_4)x_{24}+ \\
& (c_{31}+\mu_5)x_{31}+(c_{32}+\mu_6)x_{32}+c_{33}x_{33}+c_{34}x_{34}+ \\
& (c_{41}+\mu_7)x_{41}+(c_{42}+\mu_8)x_{42}+c_{43}x_{43}+c_{44}x_{44}+ \\
& (-f_1+\mu_5+\mu_7)\bar{y}_1 + (-f_2+\mu_6+\mu_8)\bar{y}_2 + (-f_3+\mu_1+\mu_3)\bar{y}_3 + (-f_4+\mu_2+\mu_4)\bar{y}_4\}
\end{aligned}
$$

Sujeito a

$$
\begin{aligned}
& x_{11}+x_{12}+x_{13}+x_{14}= 1 \\
& x_{21}+x_{22}+x_{23}+x_{24}= 1 \\
& x_{31}+x_{32}+x_{33}+x_{34}= 1 \\
& x_{41}+x_{42}+x_{43}+x_{44}= 1 \\
& x_{11}+\bar{y}_1<= 1 \\
& x_{12}+\bar{y}_2<= 1 \\
& x_{21}+\bar{y}_1<= 1 \\
& x_{22}+\bar{y}_2<= 1 \\
& x_{33}+\bar{y}_3<= 1 \\
& x_{34}+\bar{y}_4<= 1 \\
& x_{43}+\bar{y}_3<= 1 \\
& x_{44}+\bar{y}_4<= 1 \\
& x_{11}, x_{12}, x_{13}, x_{14}, x_{21}, x_{22}, x_{23}, x_{24}, x_{31}, x_{32}, x_{33}, x_{34}, x_{41}, x_{42}, x_{43}, x_{44}, \bar{y}_1, \bar{y}_2, \bar{y}_3, \bar{y}_4 \in \{0,1\}
\end{aligned}
$$

A relaxação lagrangeana pode ser, agora, decomposta em *2* subproblemas e resolvida.

$$
\begin{aligned}
v\left(\overline{UFLP}\right)_1^{LC} = Min\ \{ & c_{11}x_{11}+c_{12}x_{12}+(c_{13}+\mu_1)x_{13}+(c_{14}+\mu_2)x_{14}+ \\
& c_{21}x_{21}+c_{22}x_{22}+(c_{23}+\mu_3)x_{23}+(c_{24}+\mu_4)x_{24}+ \\
& (-f_1+\mu_5+\mu_7)\bar{y}_1 + (-f_2+\mu_6+\mu_8)\bar{y}_2\}
\end{aligned}
$$

Sujeito a

$$
\begin{aligned}
& x_{11}+x_{12}+x_{13}+x_{14}= 1 \\
& x_{21}+x_{22}+x_{23}+x_{24}= 1 \\
& x_{11}+\bar{y}_1<= 1 \\
& x_{12}+\bar{y}_2<= 1 \\
& x_{21}+\bar{y}_1<= 1 \\
& x_{22}+\bar{y}_2<= 1 \\
& x_{11}, x_{12}, x_{13}, x_{14}, x_{21}, x_{22}, x_{23}, x_{24}, \bar{y}_1, \bar{y}_2 \in \{0,1\}
\end{aligned}
$$

$$v\left(\overline{UFLP}\right)_2^{LC} = Min\ \{(c_{31}+\mu_5)x_{31}+(c_{32}+\mu_6)x_{32}+c_{33}x_{33}+c_{34}x_{34}+ (c_{41}+\mu_7)x_{41}+(c_{42}+\mu_8)x_{42}+c_{43}x_{43}+c_{44}x_{44}+ (-f_3+\mu_1+\mu_3)\bar{y}_3 + (-f_4+\mu_2+\mu_4)\bar{y}_4\}$$

Sujeito a

$x_{31}+x_{32}+x_{33}+x_{34}= 1$
$x_{41}+x_{42}+x_{43}+x_{44}= 1$
$x_{33}+\bar{y}_3<= 1$
$x_{34}+\bar{y}_4<= 1$
$x_{43}+\bar{y}_3<= 1$
$x_{44}+\bar{y}_4<= 1$
$x_{31}, x_{32}, x_{33}, x_{34}, x_{41}, x_{42}, x_{43}, x_{44}, \bar{y}_3, \bar{y}_4 \in \{0, 1\}$

3.3 Decomposição Dantzig-Wolfe e Geração de Colunas para o UFLP

Conforme mencionado no Capítulo 2, a implementação clássica de geração de colunas utiliza um problema coordenador, ou problema mestre restrito (PMR), e subproblemas geradores de colunas para o PMR. Este último, por meio de suas variáveis duais, direciona os subproblemas na busca de novas colunas que acrescentam novas informações ao próprio PMR.

A formulação apresentada em (3.9-3.10) pode ser explorada em um Algoritmo de Geração de Colunas, conforme mostrado no Capítulo 2. Assim, aplicando a decomposição Dantzig-Wolfe (DW) no problema (3.9-3.10) obtém-se o seguinte Problema Mestre Restrito:

$$v\left(\overline{UFLP}_{DW}\right)_{PL} = Min \sum_{k=1}^{K} \sum_{j \in J_k} \lambda_{jk} \left(\mathbf{c}^k \tilde{\mathbf{x}}^{jk} - \mathbf{f}^k \tilde{\bar{\mathbf{y}}}^{jk}\right) \tag{3.13}$$

Sujeito a

$$\sum_{k=1}^{K} \sum_{j \in J_k} \lambda_{jk} \left(A_k \tilde{\mathbf{w}}^{jk}\right) \leq 1 \tag{3.14}$$

$$\sum_{j \in J_k} \lambda_{jk} = 1 \quad \forall k \in \{1, ..., K\} \tag{3.15}$$

$$\lambda_{jk} \geq 0 \quad \forall k \in \{1, ..., K\}, j \in J_k \tag{3.16}$$

Em que:

- J_k : Conjunto de pontos extremos do *cluster* k.
- $\tilde{\mathbf{w}}^{jk} = \begin{bmatrix} \tilde{\mathbf{x}} \\ \tilde{\bar{\mathbf{y}}} \end{bmatrix}^{jk}$: Vetor que define pontos extremos $j \in J_k$.
- c^k e f^k : Pesos associados com vértices $v \in V_k$.
- λ_{jk} : Variáveis de decisão que correspondem a pontos extremos $j \in J_k$.

O subproblema $k, k \in \{1, ..., K\}$, tem a forma da expressão apresentada em 3.11, com o vetor de multiplicadores lagrangeanos substituídos pelo vetor de variáveis duais α associado com as restrições 3.14, sendo $\alpha \in \Re_+^{\mathrm{Q}}$ e Q a quantidade de restrições relaxadas:

$$v\left(\overline{UFLP}\right)_k^{GC} = Min\left\{\left(\mathbf{c}^k + A_k^T\alpha\right)\mathbf{x}^k - \left(\mathbf{f}^k - A_k^T\alpha\right)\bar{\mathbf{y}}^k : \mathbf{x}^k, \bar{\mathbf{y}}^k \in O_k\right\} \tag{3.17}$$

em que O_k são as restrições associadas ao subproblema k (*Cluster* k). Considerando o *PMR* em (3.13-3.16), uma nova coluna gerada pelo subproblema k é uma coluna a ser inserida no PMR se o seu custo reduzido for negativo, isto é, $v\left(\overline{UFLP}\right)_k^{GC} - \beta_k < 0$, em que β_k é a variável dual associada com a k^{esima} restrição de convexidade definida em 3.15.

A decomposição apresentada acima fornece um modo alternativo de se obter o limitante da LagClus para o $\overline{UFLP}$ ($LC_\alpha^{GC}\overline{UFLP}$), dada por:

$$v\left(LC_\alpha^{GC}\overline{UFLP}\right) = \sum_{k=1}^{K}\left\{v\left(\overline{UFLP}\right)_k^{GC}\right\} + \sum_{q=1}^{Q}\alpha_q + \sum_{j\in J} f_j \tag{3.18}$$

Conforme apresentado no Capítulo 2, o Algoritmo de Geração de Colunas necessita de um conjunto inicial de colunas composto apenas por soluções viáveis para o UFLP. O algoritmo mostrado na Figura 3.3 gera uma quantidade pré-definida de colunas, e foi utilizado nos experimentos computacionais reportados na Seção 3.4.

O processo de aplicação da geração de colunas pode ser sintetizado na Figura 3.4.

Os critérios de parada para o processo de geração de colunas são: nenhuma das novas colunas geradas tem custo reduzido negativo ou $v\left(PMR\right) \leq v\left(LC_\alpha^{GC}\overline{UFLP}\right)$. Este último critério é possível, dado que o valor da LagClus é um limitante inferior para o processo de geração de colunas e o valor do PMR é também um limitante inferior quando esse

teste é satisfeito. Com isso, o processo pode ser finalizado, mesmo tendo ainda colunas a acrescentar. A remoção de colunas acontece quando a sua quantidade ultrapassa certo valor prefixado definido como *MaxColunas*. São removidas, então, uma quantidade de colunas com os maiores custos reduzidos.

```
Rotina Inserir_Solucão_Viável_PMR (initCols)
//Seja initCols o número de colunas iniciais do PMR
  ncols = 0; // ncols é a quantidade corrente de colunas inseridas
  Enquanto (ncols < initCols) Faça
    nalocs = 0; //nalocs é a quantidade de alocações feitas
    Enquanto (nalocs < |I|) Faça
       Escolha um novo centro j aleatoriamente;
       Para i ← 1 até |I| Faça
             // O custo c_ij é considerado INFINITO quando j não pode atender a
             // demanda de i.
          Se c_ij < INFINITO Então
             Alocar o ponto de demanda i à facilidade j
                 previamente escolhida.
             nalocs = nalocs + 1;
          Fim-Se
       Fim-Para
    Fim-Enquanto
    Insira a solução factível no PMR
      //  Cada solução viável gerada pelo algoritmo acima representa K colunas no
      //  PMR, uma para cada cluster.
    ncols = ncols + K;
  Fim-Enquanto
Fim Rotina
```

Figura 3.3 - Algoritmo para inserir soluções viáveis no PMR do UFLP.

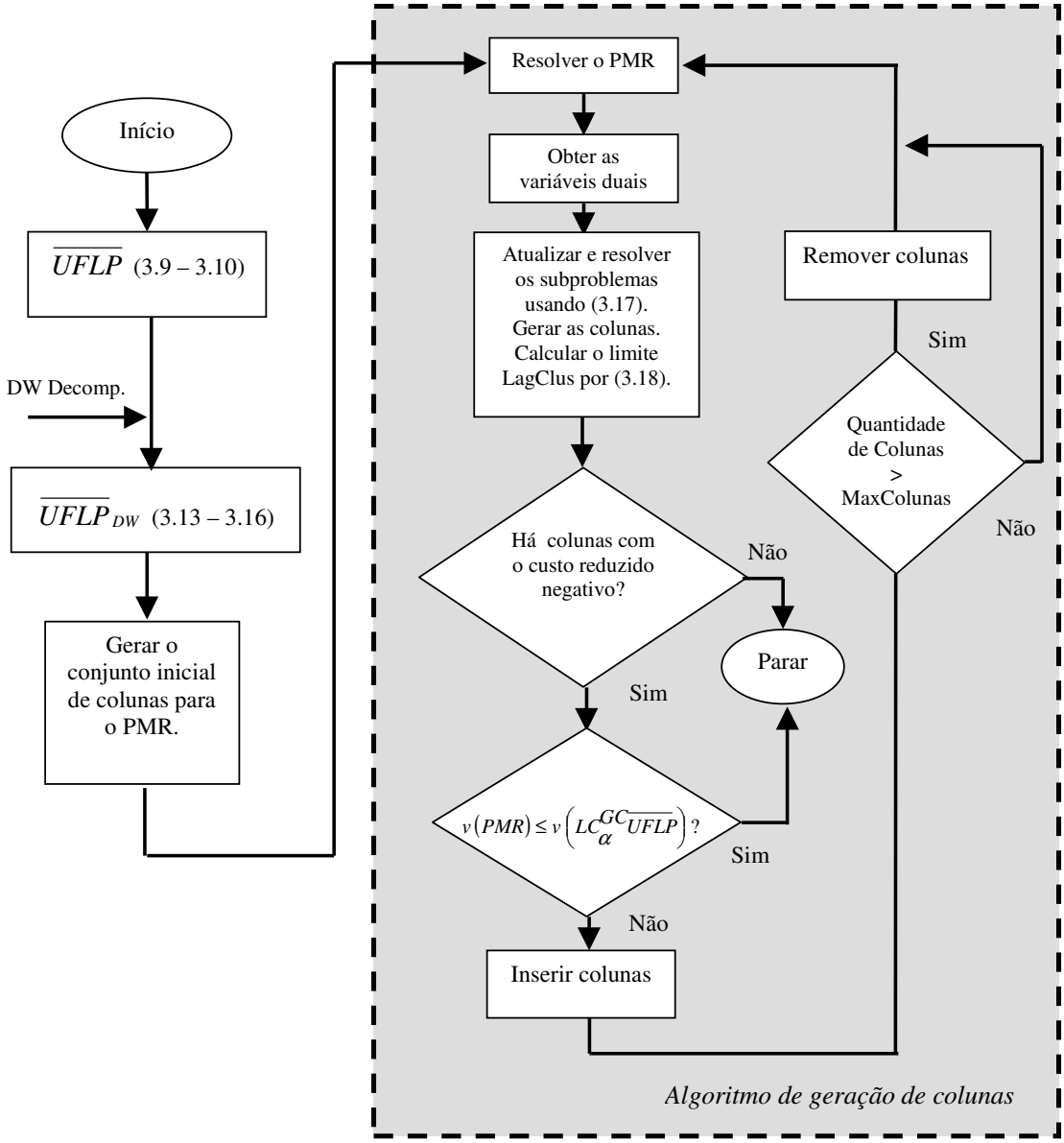


Figura 3.4 - Diagrama com os passos usados na abordagem de geração de colunas.

3.4 Resultados computacionais

Existem vários trabalhos que relatam instâncias testes para o UFLP, como, por exemplo, em Kochetov e Ivanenko (2003), Resende e Werneck (2006) e Sun (2006). Os conceitos discutidos neste Capítulo foram testados nas instâncias de Kochetov e Ivanenko (2003), por terem grande *gap* de dualidade e por serem de difícil solução para métodos baseados em relaxação de programação linear, e em algumas instâncias obtidas de *OR-Library* (BEASLEY, 1990).

Portanto, apresentam-se a seguir os principais resultados computacionais obtidos com as

técnicas descritas e aplicadas no presente Capítulo.

3.4.1 Resultados Computacionais para as Instâncias de Kochetov e Ivanenko (2003)

As instâncias de Kochetov e Ivanenko (2003) foram obtidas do site `http://www.math.nsc.ru/AP/benchmarks/UFLP/Engl/uflp_dg_eng.html`. Os autores apresentam três classes de instâncias: Gap-A, Gap-B, Gap-C. Todas as instâncias têm os valores $m = n = 100$. O custo fixo de abrir uma facilidade é 3.000. Nas instâncias do grupo Gap-A, as mais fáceis, cada cliente pode ser atribuído a dez potenciais facilidades, escolhidas aleatoriamente, com distribuição uniforme entre elas. Nas instâncias do grupo Gap-B, cada facilidade pode ser atribuída a dez potenciais clientes, escolhidos aleatoriamente, com distribuição uniforme entre eles. Nas instâncias do grupo Gap-C, as mais difíceis, cada facilidade tem dez potenciais clientes e cada cliente tem dez potenciais facilidades, escolhidos aleatoriamente, com distribuição uniforme.

Os experimentos foram realizados em um computador *Pentium IV 3 GHz*, com *1Gb* de memória *RAM*. O problema obtido com a relaxação lagrangeana, os subproblemas geradores das colunas e o *PMR* foram resolvidos com o *solver* comercial *CPLEX*, versão 7.5 (ILOG, 2001), que também foi usado para a obtenção do valor ótimo de cada instância. O particionamento dos grafos foi feito com o *METIS*.

A Tabela 3.3 mostra os valores obtidos com o CPLEX (coluna *Solução ótima*), da relaxação de programação linear (coluna *PL*) e de uma relaxação lagrangeana tradicional (coluna $v\left(L_uUFLP\right)$), que considera a relaxação das restrições definidas em (3.2), conforme indicou o Capítulo 2, com a otimização do dual lagrangeano obtida por meio do algoritmo de subgradientes. A coluna Gap PL é calculada como 100*(valor Solução ótima - PL)/(valor Solução ótima), enquanto a coluna Gap Lag é calculada como 100*(valor Solução ótima - L_uUFLP/(valor Solução ótima).

A coluna $v\left(LC_\mu\overline{UFLP}\right)$ da Tabela 3.4 mostra os valores obtidos com a relaxação lagrangeana com divisão em *clusters*. Os valores da coluna Gap LagClus foram calculados como 100*(valor Solução ótima - $v\left(LC_\mu\overline{UFLP}\right)$/(valor Solução ótima).

Para a relaxação lagrangeana tradicional e para a *LagClus* não foram implementadas heurísticas lagrangeanas que, a partir das soluções relaxadas, geram soluções viáveis para o problema primal. Nesse caso, os valores ótimos das instâncias foram utilizados para o cálculo do passo no algoritmo de subgradientes. O uso dessas melhores soluções se justifica uma vez que buscou-se investigar a qualidade dos limitantes duais.

Tabela 3.3 - Resultados da aplicação da relaxação de programação linear e relaxação lagrangeana para o UFLP em instâncias com grande *gap* de dualidade.

	CPLEX		Relaxação linear			Relaxação lagrangeana		
Instância	Solução ótima	Tempo (s)	PL	Gap PL(%)	Tempo PL (s)	$v(L_uUFLP)$	Gap Lag (%)	Tempo Lag (s)
332GapA	36154	583	26959,45	25,43	0,88	26810,08	25,84	82
432GapA	36155	1072	27223,17	24,71	0,75	27120,22	24,99	89
532GapA	36150	326	26143,15	27,68	0,83	26043,84	27,96	98
331GapB	45123	14856	34226,11	24,15	0,86	34141,59	24,34	79
431GapB	45132	18098	35031,87	22,52	0,81	34892,40	22,69	88
531GapB	45135	2287	36488,07	19,25	0,77	36432,08	19,28	133
333GapC	42147	23179	30207,90	28,33	0,86	30184,73	28,39	75
433GapC	42145	22172	30199,20	28,35	0,86	30172,98	28,42	79
533GapC	39177	3208	30200,00	22,92	0,83	30152,13	23,04	95

Tabela 3.4 - Resultados da aplicação da relaxação lagrangeana com *clusters* para o UFLP em instâncias com grande *gap* de dualidade.

Instância	Solução ótima	Número de *Clusters*	$v(LC_\mu\overline{UFLP})$	Gap LagClus (%)	Tempo (s)
332GapA	36154	2	**27491,01**	23,96	1913
332GapA	36154	4	26757,59	25,99	107
432GapA	36155	2	**27872,38**	22,91	2340
432GapA	36155	4	27063,67	25,15	127
532GapA	36150	2	**26650,84**	26,28	2244
532GapA	36150	4	25998,67	28,08	155
331GapB	45123	2	**34874,42**	22,71	2112
331GapB	45123	4	**34287,11**	24,01	135
431GapB	45132	2	**35346,06**	21,68	2010
431GapB	45132	4	34945,41	22,57	127
531GapB	45135	2	**36891,25**	18,26	1348
531GapB	45135	4	**36558,47**	19	57
333GapC	42147	2	**31273,25**	25,8	4853
333GapC	42147	4	30078,39	28,63	328
433GapC	42145	2	**31178,82**	26,02	3277
433GapC	42145	4	**30213,58**	28,31	397
533GapC	39177	2	**31152,88**	20,48	4232
533GapC	39177	4	30144,18	23,06	351

A Tabela 3.5 mostra os valores ótimos e os obtidos com a geração de colunas. Os valores mostrados na coluna Gap GC foram calculados como 100*(valor Solução ótima - $v\left(LC_{\alpha}^{GC}\overline{UFLP}\right)$)/(valor Solução ótima).

Tabela 3.5 - Resultados da aplicação da geração de colunas para o UFLP em instâncias com grande *gap* de dualidade.

		Geração de colunas				
Instância	Solução ótima	Número de *clusters*	$v\left(LC_{\alpha}^{GC}\overline{UFLP}\right)$	$v\left(\overline{UFLP}_{DW}\right)_{PL}$	Gap GC (%)	Tempo GC (s)
332GapA	36154	2	**27748,41**	**27749,26**	23,25	6743
332GapA	36154	4	**26974,35**	**26975,31**	25,39	3743
432GapA	36155	2	**28110,16**	**28110,16**	22,25	7180
432GapA	36155	4	**27250,55**	**27250,83**	24,63	2982
532GapA	36150	2	**27004,58**	**27004,58**	25,31	12342
532GapA	36150	4	**26176,45**	**26176,45**	27,59	7102
331GapB	45123	2	**34705,41**	**34706,3**	23,09	7245
331GapB	45123	4	**34240,39**	**34241,25**	24,12	4542
431GapB	45132	2	**35630,3**	**35630,31**	21,05	5791
431GapB	45132	4	**35067,49**	**35068,23**	22,3	5334
531GapB	45135	2	**36802,65**	**36803,58**	18,46	8258
531GapB	45135	4	**36664,14**	**36664,98**	18,77	3590
333GapC	42147	2	**30997,3**	**30997,81**	26,45	5247
333GapC	42147	4	**30231,52**	**30231,52**	28,27	1540
433GapC	42145	2	**31012,96**	**31013,01**	26,41	6051
433GapC	42145	4	**30218,6**	**30218,6**	28,3	1852
533GapC	39177	2	**30775,69**	**30775,69**	21,44	6677
533GapC	39177	4	**30263,21**	**30263,21**	22,75	1852

Os valores em negrito das Tabelas 3.4 e 3.5 mostram, respectivamente, onde a LagClus e a geração de colunas obtiveram melhores valores de *gap* que a PL. Merecem especial atenção as divisões do problema original em dois *clusters*, por apresentarem reduzida quantidade de arestas entre eles e melhora dos limitantes, com a penalidade nos tempos de processamento. Os valores obtidos com a geração de colunas foram melhores que os obtidos com a LagClus, e, mesmo para a divisão em quatro *clusters*, a geração de colunas foi também melhor que a relaxação de programação linear.

Considerando que o problema tratado é de minimização e a instância escolhida é de difícil solução para métodos baseados na relaxação de programação linear, os valores do problema mestre restrito e da LagClus tendem a se igualarem em um valor inferior ao do valor ótimo, conforme mostra a Figura 3.5.

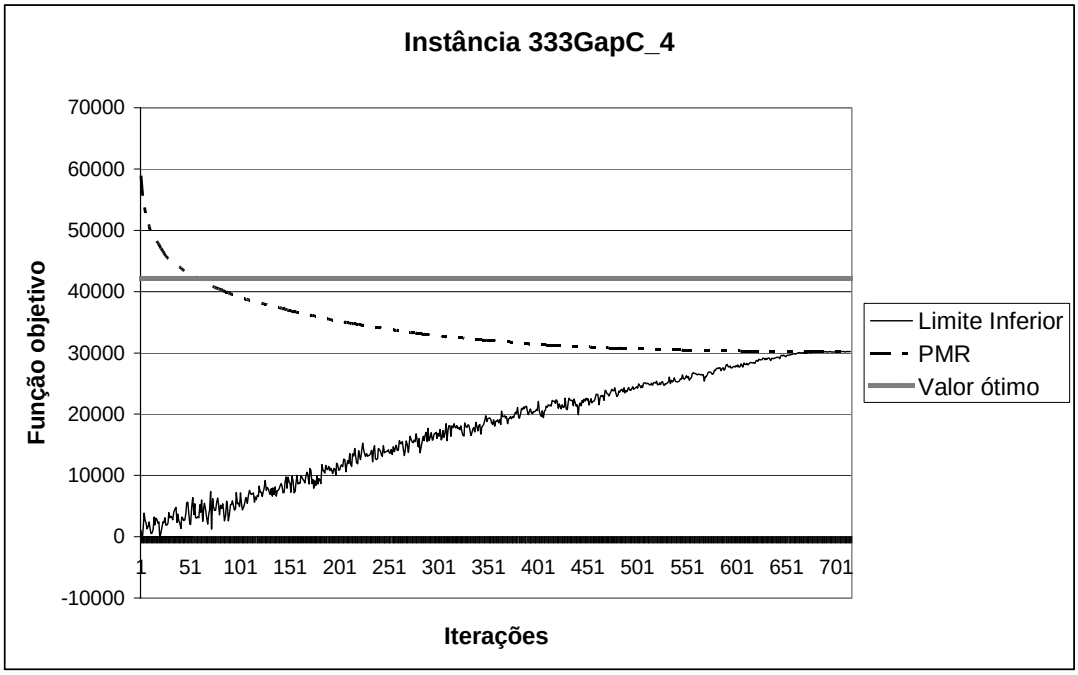


Figura 3.5 - Convergência da abordagem de geração de colunas na instância 333GapC, com divisão em quatro *clusters*.

A Tabela 3.6 foi criada para mostrar a influência da quantidade de *clusters* no tempo de processamento e nos limitantes das soluções propostas neste trabalho. O teste foi feito com a LagClus, utilizando a instância *332GapA*, que comprovou o esperado: com o aumento da quantidade de *clusters*, reduzem-se o tempo de processamento e a qualidade dos limitantes lagrangeanos.

Tabela 3.6 - Influência da quantidade de *clusters* nos resultados da LagClus em instâncias com grande *gap* de dualidade.

		Relaxação lagrangeana com divisão em *clusters*				
Instância	Solução ótima	Número de *clusters*	Número de arestas relaxadas	$v(LC_\mu\overline{UFLP})$	*Gap* LagClus (%)	Tempo (s)
332GapA	36154	2	288	27491,01	23,96	1913
332GapA	36154	4	476	26757,59	25,99	107
332GapA	36154	6	555	26746.23	26.02	15
332GapA	36154	8	612	26703,76	26,16	11

Para as instâncias de Kochetov e Ivanenko (2003), o conjunto inicial para o PMR foi de 1000 colunas, obtidas conforme o algoritmo da Figura 3.3. Sempre que o número de colunas excede 4000 (MaxColunas), são removidas as 1000 colunas de maior custo reduzido.

3.4.2 Resultados Computacionais para as Instâncias da *OR-Library*

Foram consideradas três classes de instâncias da *OR-Library*, obtidas do site `http://people.brunel.ac.uk/~mastjjb/jeb/info.html`: Cap 71 a 74, Cap 101 a 104 e Cap 131 a 134. A primeira tem $m = 50$ *e* $n = 16$, a segunda tem $m = 50$ *e* $n = 25$ e a terceira, $m = 50$ *e* $n = 50$. O custo fixo para abrir uma facilidade é diferente para cada classe. As instâncias da primeira classe são mais fáceis de resolver, seguidas pelas instâncias segunda classe e, como mais difíceis, as da terceira. Para as instâncias da *OR-Library* o conjunto inicial de colunas para o Problema Mestre definido em (3.13-3.16) é composto por 100 soluções viáveis.

As Tabelas 3.7, 3.8 e 3.9 mostram os resultados para as instâncias da *OR-Library* usando as mesmas colunas das Tabelas 3.3, 3.4 e 3.5, respectivamente.

As instâncias da *OR-Library* usadas neste trabalho são facilmente resolvidas com baixo tempo computacional por todos os métodos referenciados nas Tabelas 3.7 a 3.9. Essas são instâncias bem conhecidas, usadas como *benchmarks*, e importantes para validar algoritmos de otimização.

Tabela 3.7 - Resultados da aplicação da relaxação de programação linear e relaxação lagrangeana para o UFLP em instâncias da *OR-Library*.

Instância	CPLEX		Relaxação de programação linear			Relaxação lagrangena		
	Solução ótima	Tempo solução ótima (s)	PL	Gap PL (%)	Tempo PL (s)	$v(L_uUFLP)$	Gap Lag (%)	Tempo Lag (s)
Cap71	932615,75	0,02	932615,75	0,0	0,00	932615,75	0,0	1,83
Cap72	977799,40	0,02	977799,40	0,0	0,02	977799,40	0,0	1,89
Cap73	1010641,45	0,03	1010641,45	0,0	0,00	1010641,45	0,0	2,52
Cap74	1034976,97	0,02	1034976,97	0,0	0,00	1034976,97	0,0	2,23
Cap101	796648,44	0,02	796648,44	0,0	0,02	796648,44	0,0	2,53
Cap102	854704,20	0,01	854704,20	0,0	0,01	854704,20	0,0	2,34
Cap103	893782,12	0,02	893782,12	0,0	0,01	893782,12	0,0	2,98
Cap104	928941,75	0,02	928941,75	0,0	0,00	928941,75	0,0	2,86
Cap131	793439,57	0,05	793439,57	0,0	0,01	793439,57	0,0	7,38
Cap132	851495,33	0,05	851495,33	0,0	0,01	851495,33	0,0	7,42
Cap133	893076,71	0,05	893076,71	0,0	0,02	893076,71	0,0	9,83
Cap134	928941,75	0,05	928941,75	0,0	0,02	928941,75	0,0	10,69

A Figura 3.6 mostra a convergência da abordagem da geração de colunas nas instâncias da *OR-Library*, exemplificada na instância Cap101 com divisão em dez *clusters*. Para esse conjunto de instâncias, os valores do *PMR* e do limitante inferior tendem a serem iguais (critério de parada) no valor ótimo.

Para as instâncias da *OR-Library*, o conjunto inicial para o PMR foi de 100 colunas, obtidas conforme o algoritmo da Figura 3.3. Sempre que o número de colunas excede 4000 (MaxColunas), são removidas as 1000 colunas de maior custo reduzido.

Tabela 3.8 - Resultados da aplicação da relaxação lagrangeana com *clusters* para o UFLP em instâncias da *OR-Library*.

		Relaxação lagrangeana com divisão em *clusters*			
Instância	Solução ótima	Número de *clusters*	$v(LC_{\mu}\overline{UFLP})$	Gap LagClus (%)	Tempo LagClus (s)
Cap71	932615,75	8	932615,75	0,0	1,39
Cap72	977799,40	8	977799,31	0,0	1,86
Cap73	1010641,45	8	1010641,44	0,0	2,27
Cap74	1034976,98	8	1034976,98	0,0	2,34
Cap101	796648,44	10	796648,44	0,0	2,67
Cap102	854704,20	10	854704,20	0,0	3,34
Cap103	893782,12	10	893782,12	0,0	3,63
Cap104	928941,75	10	928941,75	0,0	3,34
Cap131	793439,57	20	793439,57	0,0	7,67
Cap132	851495,33	20	851495,33	0,0	8,2
Cap133	893076,71	20	893076,71	0,0	8,41
Cap134	928941,75	20	928941,56	0,0	7,98

Tabela 3.9 - Resultados da aplicação da geração de colunas para o UFLP em instâncias da *Or-Library*.

		Geração de colunas				
Instância	Solução ótima	Número de *clusters*	$v\left(LC_{\alpha}^{GC}\overline{UFLP}\right)$	$v\left(\overline{UFLP}_{DW}\right)_{PL}$	Gap GC (%)	Tempo GC (s)
Cap71	932615,75	8	932615,75	932615,75	0,0	0,25
Cap72	977799,40	8	977799,40	977799,40	0,0	0,33
Cap73	1010641,45	8	1010641,45	1010641,45	0,0	0,75
Cap74	1034976,98	8	1034976,98	1034976,98	0,0	2,05
Cap101	796648,44	10	796648,44	796648,44	0,0	0,67
Cap102	854704,20	10	854704,20	854704,20	0,0	1,13
Cap103	893782,12	10	893782,12	893782,12	0,0	4,03
Cap104	928941,75	10	928941,75	928941,75	0,0	10,53
Cap131	793439,57	20	793439,57	793439,57	0,0	2,95
Cap132	851495,33	20	851495,33	851495,33	0,0	6,95
Cap133	893076,71	20	893076,71	893076,71	0,0	10,73
Cap134	928941,75	20	928941,75	928941,75	0,0	32,88

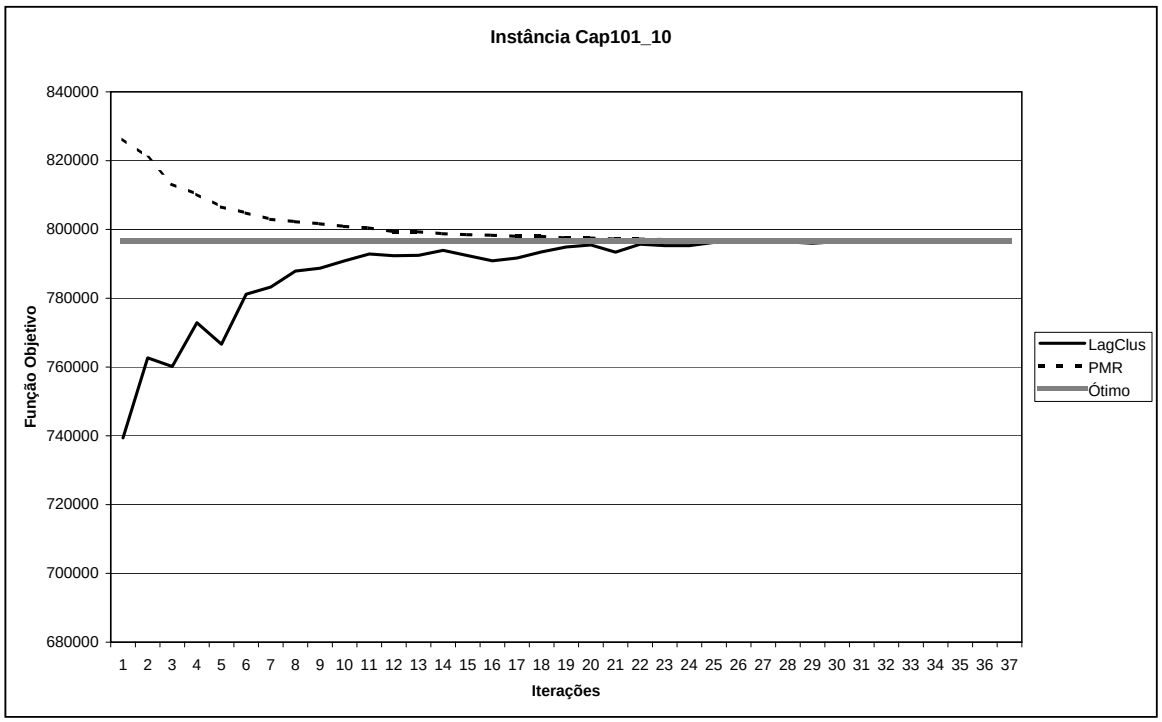


Figura 3.6 - Convergência da abordagem de geração de colunas em instância da OR-Library.

3.5 Considerações finais

Este Capítulo apresentou a relaxação lagrangeana baseada em *clusters* e um método de geração de colunas para o UFLP aplicados a instâncias de difícil solução para métodos baseados em relaxação de programação linear e a instâncias de fácil solução obtidas da *OR-Library*.

Duas relaxações foram ainda consideradas para a resolução deste modelo: a relaxação de programação linear e a lagrangeana tradicional. Essa última apresentou limitantes de qualidade inferior comparados aos da primeira.

A Tabela 3.5 mostra que o Algoritmo de Geração de Colunas apresentou-se como a melhor estratégia de solução se comparado à LagClus e as demais relaxações, por apresentar limites duais de melhor qualidade, apesar dos tempos computacionais serem maiores.

O próximo Capítulo apresenta algumas técnicas alternativas para a solução do problema probabilístico de localização-alocação de máxima cobertura, incluindo a LagClus e o algoritmo de geração de colunas aplicados sobre a modelagem do problema em grafo de cobertura.

4 PROBLEMA PROBABILÍSTICO DE LOCALIZAÇÃO-ALOCAÇÃO DE MÁXIMA COBERTURA

Conforme visto no Capítulo 2, os problemas de localização têm como objetivo localizar facilidades para atender aos usuários que estão espacialmente distribuídos. As facilidades são centros que fornecem algum tipo de serviço.

Entre esses problemas, o de Localização de Máxima Cobertura (PLMC) tem sido consideravelmente tratado na literatura desde a sua formulação feita por Church e ReVelle (1974). Esse problema busca obter a configuração para localizar uma quantidade pré-definida de facilidades que atenda o maior número de indivíduos de uma população, considerado um específico raio de cobertura. Não se busca com este modelo atender toda a população, mas oferecer o máximo de atendimento, considerando os recursos disponíveis.

Ainda conforme visto no Capítulo 2, vários modelos aplicados a uma grande faixa de problemas, nos setores público e privado, são extensões dessa formulação, adaptados para melhor representar a realidade.

Em muitas dessas aplicações práticas, a distância ou o tempo entre os pontos de demanda e as facilidades constituem fatores importantes para estabelecer o nível de serviço oferecido aos usuários. Por outro lado, sabe-se que as chegadas dos usuários em sistemas de atendimento, como hospitais e bancos, são regidas por processos aleatórios o que, muitas vezes, proporcionam um congestionamento e a geração de filas (FOGLIATTI; MATTOS, 2007). Sistemas que apresentam tal comportamento indicam que o nível de serviço prestado aos usuários não pode ser medido levando-se em consideração apenas a cobertura dos pontos de demanda, mas também os atributos de fila como o comprimento e o tempo de espera. Sendo assim, os centros devem ser localizados de tal maneira que os usuários cheguem dentro de um tempo aceitável e também que, uma vez na fila, o tempo de espera não seja maior que um limite máximo e/ou que o comprimento da mesma não seja maior que um valor máximo (MARIANOV; SERRA, 1998).

O congestionamento ocorre quando um centro não é capaz de atender, simultaneamente, a todas as solicitações de serviços que lhe são feitas. Os modelos tradicionais que tratam desse problema adicionam uma restrição de capacidade que força a demanda por serviço, normalmente constante no tempo e igual a uma média, a ser menor do que a máxima capacidade do centro. Essa abordagem não considera a natureza dinâmica do congestionamento e trata o problema de forma determinística. Isso faz com que o modelo, dependendo de como a restrição seja obtida, tenha servidores ociosos ou não tenha a capacidade de atender todas as demandas (MARIANOV; SERRA, 1998; MARIANOV; SERRA,

2001).

Considerando então essa aleatoriedade no processo de chegada e atendimento, Marianov e Serra (1998) propuseram modelos matemáticos em que a estocasticidade da demanda é explicitamente considerada nas restrições de capacidade, que, em vez de serem limitadas a um máximo, os autores definem um limite mínimo para a qualidade dos seus serviços. Essa qualidade é refletida no tempo de espera ou na quantidade de pessoas que aguardam o atendimento.

Esses autores definiram então o Problema Probabilístico de Localização-Alocação de Máxima Cobertura (PPLAMC), que busca localizar uma dada quantidade de facilidades com um ou vários servidores, de modo que a população, a uma distância padrão do centro, seja servida adequadamente, isto é, que ninguém fique na fila por um período maior que um dado tempo limite ou que um usuário, ao chegar ao centro, não encontre um número de outros clientes acima do previsto, com probabilidade maior ou igual a dado valor definido *a priori*. Os autores trabalharam com postos de saúde, modelos de fila do tipo $M/M/1/\infty/FIFO$ e $M/M/m/\infty/FIFO$ (LARSON; ODONI, 1981) e taxas de chegadas a cada centro j distribuídas conforme uma Poisson com taxa ω_j e tempo de atendimento exponencialmente distribuído com taxa ϕ_j para cada servidor.

O propósito deste Capítulo é o de examinar o PPLAMC proposto por Marianov e Serra (1998), para um servidor por centro, e apresentar algumas soluções. Limitantes lagrangeanos são obtidos com a relaxação lagrangeana tradicional e a LagClus. São utilizadas, também, as metaheurísticas Algoritmo Genético Construtivo (AGC) e *Clustering Search* (CS). Além disto, considerando a idéia da LagClus e a relação entre o dual lagrangeano e a decomposição Dantzig-Wolfe, resolve-se ainda o PPLAMC com o Algoritmo de Geração de Colunas. Estes métodos são testados em instâncias encontradas na literatura com até 818 vértices.

A LagClus, conforme mostrada no Capítulo 2, utiliza uma representação do problema por meio de um grafo de conflitos que é particionado em *clusters*. Apresenta-se neste capítulo a LagClus também sobre o problema representado por um grafo de cobertura. O particionamento deste último produz um número menor de restrições a serem relaxadas que o grafo de conflitos. Resultados computacionais mostram que a LagClus e o Algoritmo de Geração de Colunas com o grafo cobertura apresentam bons limitantes.

4.1 Formulações para o PPLAMC

O modelo tradicional PLMC proposto por Church e ReVelle (1974) não pode ser usado para tratar as restrições de congestionamento, pois não contém variáveis de alocação,

o que impede de computar as solicitações de serviços que chegam a um centro, e, conseqüentemente, de determinar quando ocorre um congestionamento.

A modelagem matemática proposta para o PPLAMC foi escrita como um problema tipo p-Medianas, modificado para comportar as variáveis de localização e alocação, tendo como objetivo maximizar a população coberta, considerando uma determinada quantidade de centros de atendimento (facilidades).

Seja I o conjunto dos pontos de demanda a serem alocados e N_i o conjunto de localizações candidatas que estão dentro de uma dada distância do ponto i. Formalmente, as alocações são representadas pelas variáveis binárias x_{ij}, $\forall i \in I$ e $j \in N_i$. Com isto, $x_{ij} = 1$, se o ponto de demanda i for alocado ao centro j, $x_{ij} = 0$, caso contrário. As localizações são representadas pelas variáveis binárias y_j, com $y_j = 1$, se o centro j for selecionado e $y_j = 0$, caso contrário. Todo ponto de demanda é um potencial centro de atendimento.

Assim, a formulação do PPLAMC pode ser escrita (MARIANOV; SERRA, 1998):

$$v(PPLAMC) = Max \left\{ \sum_{i \in I} \sum_{j \in N_i} a_i x_{ij} \right\} \tag{4.1}$$

Sujeito a

$$x_{ij} \leq y_j \quad \forall i \in I \text{ e } j \in N_i \tag{4.2}$$

$$\sum_{j \in N_i} x_{ij} \leq 1 \quad \forall i \in I \tag{4.3}$$

$$\sum_{i \in I} f_i x_{ij} \leq \phi_j \sqrt[b+2]{1-\varphi} \quad \forall j \in N_i \tag{4.4}$$

$$\sum_{i \in I} f_i x_{ij} \leq \phi_j + \frac{1}{\tau} \ln(1-\varphi) \quad \forall j \in N_i \tag{4.5}$$

$$\sum_{i \in I} y_i = p \tag{4.6}$$

$$y_j \text{ e } x_{ij} \in \{0,1\} \quad \forall i \in I \text{ e } j \in N_i \tag{4.7}$$

- a_i representa a população total do ponto de demanda i;
- b é o número máximo de usuários na fila com probabilidade de, no mínimo, φ;
- τ é o tempo máximo de espera na fila com probabilidade de, no mínimo, φ;
- f_i é a taxa de chegada a um centro dos usuários provenientes de um ponto de demanda i conforme um processo de Poisson;
- ϕ_j é a taxa média de atendimento em que o tempo de atendimento é exponencialmente distribuído;
- p é o número de facilidades a serem localizadas.

A função objetivo descrita em (4.1) indica que a população total coberta pelos p centros deve ser a maior possível. As restrições definidas em (4.2) garantem que só é possível alocar um ponto de demanda i a um centro j se houver um centro em j. As restrições descritas em (4.3) garantem que cada ponto de demanda deve ser alocado a, no máximo, um centro. As restrições descritas em (4.4) e (4.5) estão associadas à questão do comprimento máximo da fila e ao tempo máximo de atendimento, respectivamente. Utiliza-se uma delas, dependendo da restrição que se quer tratar. As definidas por (4.4) garantem que, com probabilidade $\geq \varphi$, cada centro tenha no máximo b pessoas na fila. Por outro lado, as definidas em (4.5) garantem que, com probabilidade $\geq \varphi$, o tempo de atendimento em cada centro seja de no máximo τ. A restrição definida em (4.6) garante que p centros serão selecionados, e as restrições descritas em (4.7), que todas as variáveis são binárias.

Para escrever as restrições (4.4) e (4.5), Marianov e Serra (1998) consideraram o sistema de filas $M/M/1/\infty/FIFO$, em que as solicitações de serviços de cada ponto de demanda i acontecem de acordo com um processo de Poisson com taxa f_i, o tempo de atendimento exponencialmente distribuído, com um servidor, sem limite de capacidade e disciplina de fila do tipo "o primeiro a chegar é o primeiro a ser atendido"(*first in - first out*). Considerando que os usuários vindos de vários pontos de demanda i chegam a um centro j de maneira superposta, a taxa ω_j atribuída a esse centro é definida como uma superposição de processos de Poisson:

$$\omega_j = \sum_{i \in I} f_i x_{ij} \tag{4.8}$$

Para que se tenha equilíbrio no processo de atendimento dos centros, faz-se necessário que $\phi_j > \omega_j$.

Se o estado k for definido como sendo k usuários no sistema (em atendimento ou na fila), o diagrama de transição de estado é o mostrado na Figura 4.1. Isto é, estado k = 0 significa sistema em espera; k = 1, significa um usuário sendo atendido; para k = 2, tem-se um usuário sendo atendido e um na fila, e assim por diante.

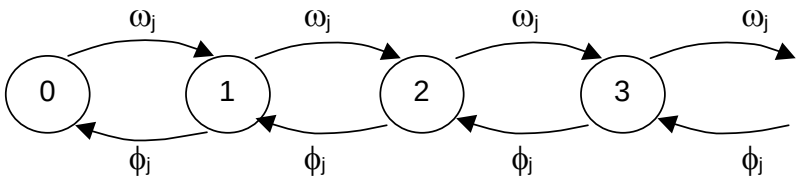


Figura 4.1 - Diagrama de transição de estado.

Deseja-se que a probabilidade de um usuário não encontrar mais do que b pessoas na fila seja, no mínimo, φ. Se p_k representar a probabilidade de estar no estado k, então esse requisito pode ser escrito como:

$$p_0 + p_1 + p_2 + p_3 + \ldots + p_{b+1} \geq \varphi \tag{4.9}$$

Considerando as equações de equilíbrio para o sistema de filas $M/M/1/\infty/FIFO$, obtém-se a seguinte expressão:

$$p_k = (1 - \rho_j)\,\rho_j^k \tag{4.10}$$

Em que $\rho_j = \frac{\omega_j}{\phi_j}$. De (4.9) e (4.10), tem-se:

$$(1 - \rho_j) + (1 - \rho_j)\,\rho_j + (1 - \rho_j)\,\rho_j^2 + \ldots + (1 - \rho_j)\,\rho_j^{b+1} \geq \varphi, \text{ ou}$$

$$(1 - \rho_j) \sum_{k=0}^{b+1} \rho_j^k \geq \varphi$$

Da soma dos elementos de uma progressão geométrica finita, tem-se:

$$(1 - \rho_j)\,\frac{\left(1-\rho_j^{b+2}\right)}{(1-\rho_j)} \geq \varphi$$

$$1 - \rho_j^{b+2} \geq \varphi$$

$$\rho_j^{b+2} \leq 1 - \varphi$$

$$\rho_j \leq \sqrt[b+2]{1-\varphi}$$

Como $\rho_j = \frac{\omega_j}{\phi_j}$, pode-se escrever:

$$\omega_j = \phi_j \sqrt[b+2]{1-\varphi} \tag{4.11}$$

Das equações (4.8) e (4.11), tem-se:

$\sum_{i \in I} f_i x_{ij} \leq \phi_j \sqrt[b+2]{1-\varphi} \quad \forall j \in N_i$, que é a equação (4.4). O desenvolvimento da equação (4.5) pode ser encontrado também em Marianov e Serra (1998).

O lado direito das equações (4.4) e (4.5) são constantes, calculados para ϕ_j, φ, b e τ, definidos *a priori*. Simplificando, essas restrições podem ser substituídas, respectivamente, pelos dois conjuntos de restrições a seguir, em que $Z^j_{\phi b \varphi} = \phi_j \sqrt[b+2]{1-\varphi}$ e $Z^j_{\phi \tau \varphi} = \phi_j + \frac{1}{\tau}\ln(1-\varphi)$.

$$\sum_{i \in I} f_i x_{ij} \leq Z^j_{\phi b \varphi} \quad \forall j \in N_i \tag{4.12}$$

$$\sum_{i \in I} f_i x_{ij} \leq Z^j_{\phi \tau \varphi} \quad \forall j \in N_i \tag{4.13}$$

O modelo proposto por Marianov e Serra (1998), descrito anteriormente, pode ser alterado para comportar restrições de adjacência ou de conflitos. Considerando então o complemento das variáveis de localização, ou seja, $\bar{y}_j = 1 - y_j$, o modelo $\overline{PPLAMC}$ pode ser escrito:

$$v(\overline{PPLAMC}) = Max \left\{ \sum_{i \in I} \sum_{j \in N_i} a_i x_{ij} \right\} \tag{4.14}$$

Sujeito a (4.3) e (4.12) ou (4.13), e

$$x_{ij} + \bar{y}_j \leq 1 \quad \forall i \in I \text{ e } j \in N_i \tag{4.15}$$

$$\sum_{i \in I} \bar{y}_i = |N| - p \tag{4.16}$$

$$\bar{y}_j \text{ e } x_{ij} \in \{0,1\} \quad \forall i \in I \text{ e } j \in N_i \tag{4.17}$$

Em que $N = \cup_{i \in I} N_i$.

Outra alteração que pode ser feita no modelo definido por Marianov e Serra (1998) é considerar as restrições (4.2) e de capacidade em uma única restrição (MURRAY; GERRARD, 1997). Desta forma, uma nova formulação ($PPLAMC^{MG}$) pode ser obtida, conforme mostrado a seguir:

$$v(PPLAMC^{MG}) = Max \left\{ \sum_{i \in I} \sum_{j \in N_i} a_i x_{ij} \right\} \tag{4.18}$$

Sujeito a (4.3), (4.6), (4.7) e

$$\sum_{i \in I} f_i x_{ij} \leq Z_{\phi b \, \varphi} y_j \quad \forall j \in N_i \tag{4.19}$$

$$\sum_{i \in I} f_i x_{ij} \leq Z_{\phi \tau \, \varphi} y_j \quad \forall j \in N_i \tag{4.20}$$

As restrições (4.19) substituem as restrições (4.2) e (4.12), pois exigem que somente seja possível alocar um ponto de demanda i a um centro j se houver um centro em j, como fazem as restrições (4.2), além de imporem restrições de capacidade, como fazem as definidas em (4.12). Do mesmo modo, as restrições (4.20) substituem as restrições (4.2) e (4.13).

Considerando as instâncias de grande porte (324 e 818 pontos de demanda) propostas por Corrêa e Lorena (2006) e Corrêa et al. (2008) disponíveis em `http://www.lac.inpe.br/~lorena/instancias.html`, o CPLEX 10 (ILOG, 2006) não consegue resolver todas as instâncias, parando algumas vezes por falta de memória. Conseqüentemente buscam-se soluções alternativas, como heurísticas, relaxações e métodos de decomposição. Essas três formulações acima discutidas serão usadas nas soluções propostas para o PPLAMC, conforme mostrado a seguir.

4.2 Relaxação Lagrangeana

Para aplicar a relaxação lagrangeana ao problema proposto por Marianov e Serra (1998), utilizou-se a formulação $PPLAMC^{MG}$. As restrições de alocação (4.3) foram relaxadas no sentido lagrangeano e incorporadas na função objetivo. Portanto, o problema $L_uPPLAMC^{MG}$ pode ser assim descrito:

$$v\left(L_uPPLAMC^{MG}\right) = Max \sum_i \sum_j \left(a_i - u_i\right) x_{ij} + \sum_i u_i \tag{4.21}$$

Sujeito a (4.6), (4.7) e (4.19) ou (4.20).

O problema $L_uPPLAMC^{MG}$ pode ser decomposto em $|I|$ subproblemas. Em cada um deles, a variável de localização y_j pode ser igual a zero ou um. Se for zero, todas as variáveis de alocação x_{ij} serão também zero e seus correspondentes valores na formulação tornam-se zero. Quando y_j tem o valor um, os subproblemas tornam-se o problema da mochila 0-1 e podem ser resolvidos independentemente. Se forem tomados os p melhores resultados, a restrição de cardinalidade (4.6) estará sendo considerada implicitamente (PIRKUL; SCHILLING, 1991; LORENA; SENNE, 2003).

Assim, o problema original pode ser reescrito com um conjunto de subproblemas:

Para $j = 1, 2, ..., |I|$,

$$v\left(Knap^j\right) = Max \sum_{i \in I} \left(a_i - u_i\right) x_{ij} \tag{4.22}$$

Sujeito a

$$\sum_{i \in I} f_i x_{ij} \leq Z_{\phi b \varphi} \tag{4.23}$$

$$\sum_{i \in I} f_i x_{ij} \leq Z_{\phi \tau \varphi} \tag{4.24}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i \in I \text{ e } j \in N_i \tag{4.25}$$

Cada problema é resolvido usando-se o algoritmo de Horowitz e Sahni (MARTELLO; TOTH,

1990). Esse algoritmo ofereceu bons resultados, embora tenha sido desenvolvido para problemas inteiros de mochila.

Seja J o conjunto dos p maiores $v(Knap^j), j \in I$. O valor da relaxação Lagrangeana é dado por:

$$v\left(L_u PPLAMC^{MG}\right) = \sum_{j \in J} v\left(Knap^j\right) + \sum_{i \in I} u_i \tag{4.26}$$

A otimização do dual Lagrangeano é obtida por meio do algoritmo de subgradientes (PARKER; RARDIN, 1988; NARCISO; LORENA, 1999; LORENA; SENNE, 2003).

As soluções x^u obtidas não são necessariamente viáveis, porém o conjunto J indica as facilidades que podem ser usadas para se obter soluções viáveis (v^f, x^f). Para alocar os pontos de demanda ao conjunto de facilidades identificadas, busca-se a solução aproximada para a seguinte variante do Problema de Atribuição Generalizado, do inglês *Generalized Assignment Problem* (GAP), com desigualdades nas restrições (4.24), sendo o operador relacional *menor ou igual* (LEGAP):

$$Max \sum_{i \in I} \sum_{j \in J} a_i x_{ij}^f \tag{4.27}$$

Sujeito a

$$\sum_{j \in J} x_{ij}^f \leq 1 \quad i \in I \tag{4.28}$$

$$\sum_{i \in I} f_i x_{ij}^f \leq Z_{\phi b \varphi} \quad j \in J \tag{4.29}$$

$$\sum_{i \in I} f_i x_{ij}^f \leq Z_{\phi \tau \varphi} \quad j \in J \tag{4.30}$$

$$x_{ij}^f \in \{0, 1\}, \quad i \in I \quad j \in J \tag{4.31}$$

As soluções primais foram obtidas por meio do algoritmo *MTHG*, de Martello e Toth (1990), adaptado para contemplar as desigualdades, pois ao contrário do *GAP*, o *LEGAP*

sempre admite solução viável, e melhoradas pelo algoritmo localização-alocação descrito em Lorena e Pereira (2002). Este último leva a obter aumentos no valor da função objetivo (v^f) correspondentes a uma solução primal (x^f), substituindo-se uma facilidade por um vértice do mesmo *cluster* e recalculando-se o atendimento dos pontos de demanda após essa troca, pois a configuração de cobertura desses pontos pode ser modificada após essa substituição, como pode ser visto na Figura 4.2. O algoritmo de melhoria de soluções primais é apresentado na Figura 4.3.

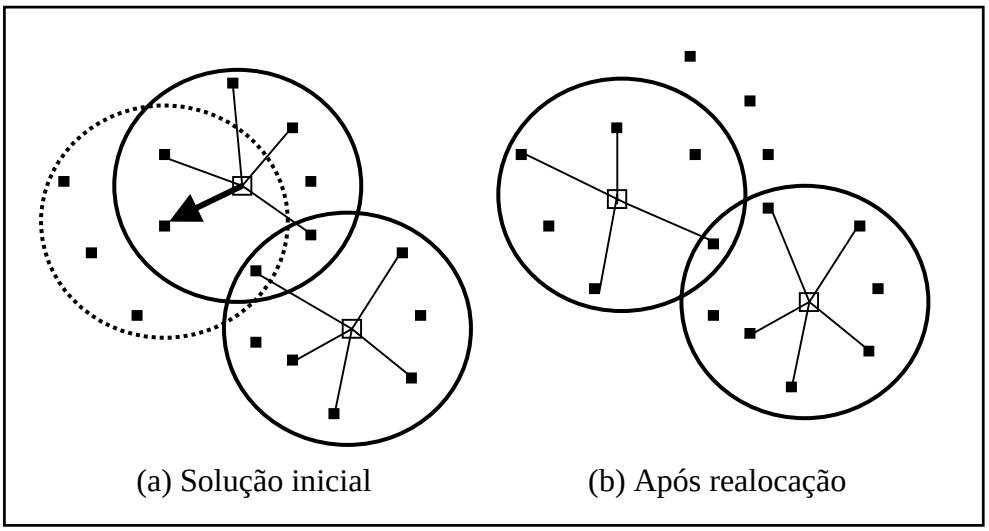


Figura 4.2 - Realocação de facilidade por ponto de demanda.

Fonte: adaptado de Lorena e Pereira (2002).

```
Enquanto (v^f aumenta)
    Para k = 1, ..., p
        Troque o vértice facilidade por um vértice não-facilidade do cluster C^k.
        Calcule o valor v correspondente à melhor realocação.
        Se v > v^f
            Atualize a facilidade do cluster C^k.
            Faça v^f = v.
        Fim-Se;
    Fim-Para;
Fim-Enquanto;
```

Figura 4.3 - Algoritmo de melhoria de soluções primais.

Fonte: adaptado de Lorena e Pereira (2002).

4.3 Relaxação Lagrangeana com *Clusters* a partir de Grafos de Conflitos e de Cobertura

A LagClus pode ser aplicada ao PPLAMC a partir de sua modelagem por grafo de conflitos ou de cobertura. O grafo de conflitos pode ser obtido a partir do conjunto de restrições de adjacências definidos em (4.3) e (4.15) da formulação $\overline{PPLAMC}$. O grafo de cobertura é obtido do modelo original de Marianov e Serra (1998).

A seguir, é apresentado um exemplo do grafo de conflitos para o $\overline{PPLAMC}$ e do grafo de cobertura para o PPLAMC.

4.3.1 Grafo de Conflitos para o $\overline{PPLAMC}$

Sejam os dados da Tabela 4.1 que indicam cinco pontos de demanda com suas respectivas coordenadas de localização. Admitindo que cada ponto de demanda possa ser um centro e que um ponto i estará coberto por um centro j se esse estiver a uma distância euclidiana menor ou igual a 4, a coluna *Cobertura* apresenta os pontos cobertos por cada um dos possíveis centros. Levando-se em consideração o modelo mostrado para o $\overline{PPLAMC}$, as restrições definidas em (4.3) e (4.15) podem ser representadas por um grafo de conflitos (veja Figura 4.4).

Tabela 4.1 - Exemplo de um problema de máxima cobertura com raio de cobertura igual a 4.

Pontos	Coordenadas		Demanda	Capacidade	Cobertura
	X	Y			
1	4	8	7	10	1, 2, 4, 5
2	7	10	6	10	1, 2, 3
3	8	7	6	10	2, 3, 4
4	5	5	4	10	1, 3, 4, 5
5	2	6	3	10	1, 4, 5

Considerando agora o segundo passo da LagClus, suponha que o grafo de conflitos tenha sido particionado em 2 *clusters*, conforme mostra a Figura 4.4, e que as cliques tenham sido colapsadas antes do particionamento. Nessa figura, as linhas mais espessas mostradas a direita indicam quais restrições devem ser relaxadas (nesse exemplo, restrições definidas conforme (4.15)).

Ao considerar o $\overline{PPLAMC}$ e o particionamento mostrado na Figura 4.4, além das restrições de conflitos já identificadas previamente para serem relaxadas, algumas restrições do tipo (4.12) ou (4.13) e a restrição (4.16) também devem ser relaxadas.

Ao escrever o modelo matemático todo para o exemplo, percebe-se que onze restrições são relaxadas: seis estão mostradas em negrito na Figura 4.4, a restrição (4.16) e quatro restrições do tipo (4.12) ou (4.13), pois as variáveis x_{ij} estão em diferentes *clusters* (por exemplo, a restrição de capacidade para um centro colocado no ponto 1: $7x_{1\_1} + 6x_{2\_1} + 4x_{4\_1} + 3x_{5\_1} \leq 10$).

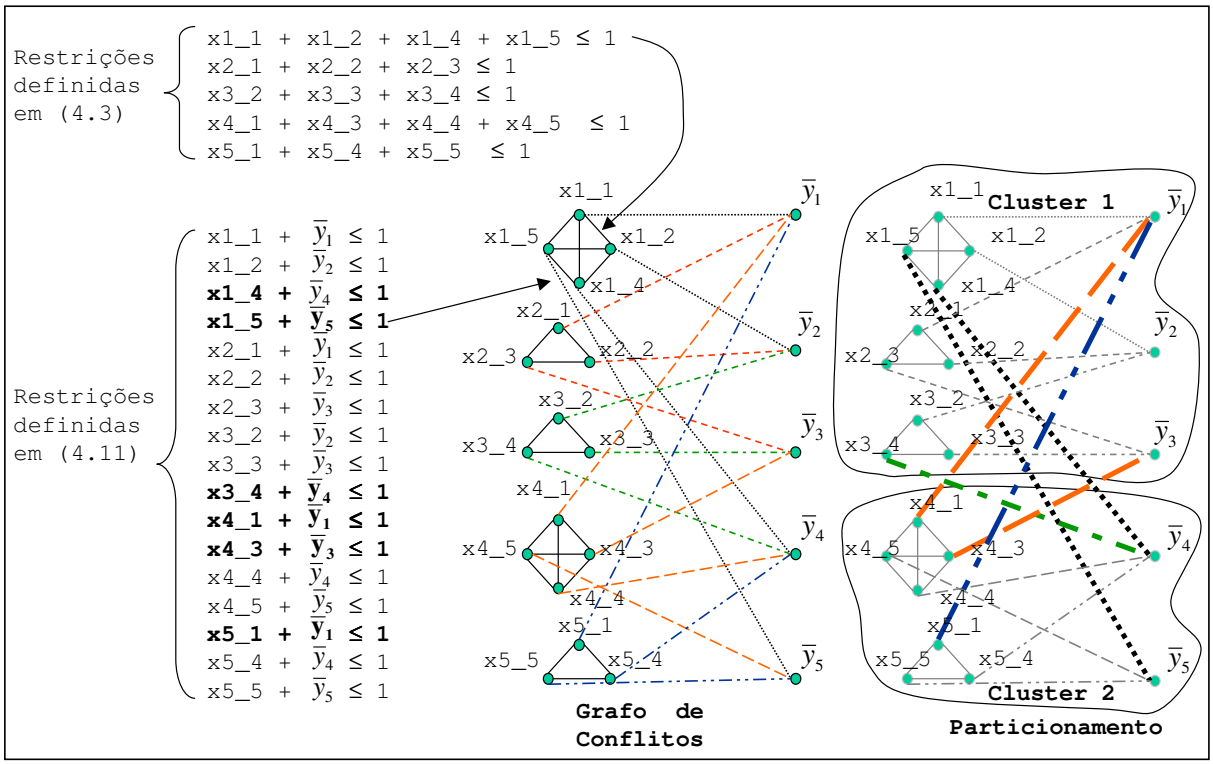


Figura 4.4 - Exemplo de um grafo de conflitos para o $\overline{PPLAMC}$.

Resultados computacionais mostraram que a decomposição do grafo de conflitos do PPLAMC em *clusters*, relaxa muitas restrições o que gera limitantes duais de baixa qualidade. Isso motivou o estudo do particionamento dos grafos de cobertura ao invés dos grafos de conflitos. O grafo de cobertura é obtido a partir da primeira formulação do PPLAMC proposta por Marianov e Serra (1998), como será mostrado a seguir.

4.3.2 Grafo de Cobertura para o PPLAMC

Um grafo de cobertura $G = (V,E)$ apresenta um conjunto V de vértices formados pelos locais candidatos e um conjunto E de arestas que indica a cobertura de cada ponto de demanda, conforme tratado no Capítulo 2. Esse grafo também apresenta *clusters* de vértices, logo a LagClus pode ser aplicada.

A partir dos dados apresentados na Tabela 4.1, sabe-se que os conjuntos de localizações candidatas de cada ponto são: $N_1 = \{1, 2, 4, 5\}$, $N_2 = \{1, 2, 3\}$, $N_3 = \{2, 3, 4\}$, $N_4 = \{1, 3, 4, 5\}$ e $N_5 = \{1, 4, 5\}$. Sendo assim, a Figura 4.5 apresenta a cobertura de todos os pontos e, por fim, o grafo de cobertura do problema.

Particionando o grafo de cobertura em dois *clusters* como realizado na Figura 4.5, ou seja, os pontos 1, 2 e 3 no *cluster* 1 e os pontos 4 e 5 no *cluster* 2, e considerando a formulação de Marianov e Serra (1998), as seguintes restrições devem ser relaxadas: $x_{1\_1} + x_{1\_2} + x_{1\_4} + x_{1\_5} \leq 1$, $x_{3\_2} + x_{3\_3} + x_{3\_4} \leq 1$, $x_{4\_1} + x_{4\_3} + x_{4\_4} + x_{4\_5} \leq 1$ e $x_{5\_1} + x_{5\_4} + x_{5\_5} \leq 1$, além da restrição (4.6). Isso resulta em apenas cinco restrições relaxadas.

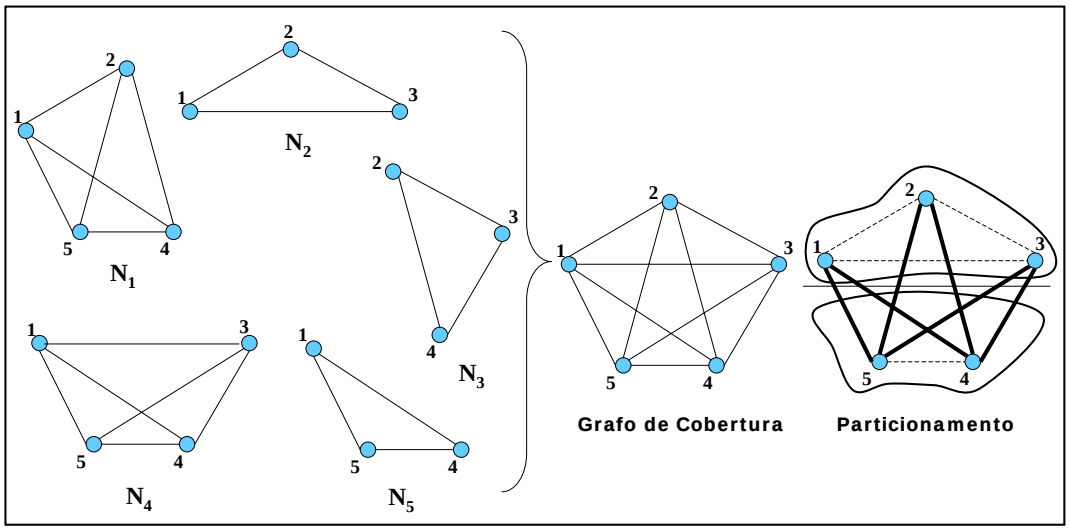


Figura 4.5 - Exemplo de um grafo de cobertura para o PPLAMC.

Para comparar o número de restrições relaxadas por cada um dos métodos, grafos de conflitos e de cobertura, foram realizados experimentos com três conjuntos de instâncias propostos na literatura. O primeiro (C_1) foi proposto por Marianov e Serra (1998) e contém instâncias com 30 pontos, descritos na Tabela 4.3; o segundo e o terceiro (C_2 e C_3) possuem, respectivamente, 324 e 818 pontos, foram propostos por Corrêa e Lorena (2006) e Corrêa et al. (2007) e estão disponíveis em `http://www.lac.inpe.br/~lorena/instancias.html`. O particionamento do grafo foi realizado pela heurística *METIS* (Karypis e Kumar, 1998). Os experimentos foram realizados variando-se o número de *clusters* para uma melhor comparação. Os resultados estão mostrados na Tabela 4.2.

Analisando a Tabela 4.2, note que no grafo de conflitos particiona-se um número menor de arestas que no grafo de cobertura. Por outro lado, o número de restrições relaxadas

no modelo com restrições de conflitos é maior que o do outro modelo. Ao considerar os valores médios, o modelo $\overline{PPLAMC}$ relaxa aproximadamente 30 vezes mais restrições que o modelo PPLAMC. Esse comportamento justifica o uso da LagClus com o grafo de cobertura e a formulação original de Marianov e Serra (1998), pois, em geral, essa relaxação é dependente do número de restrições relaxadas, ou seja, quanto maior é esse número, mais "fraca" se torna a LagClus (RIBEIRO, 2007). Além disso, as restrições de capacidade não são relaxadas no caso do grafo de cobertura, pois estão definidas para cada centro (ponto).

Tabela 4.2 - Número de restrições relaxadas na LagClus usando grafos de conflitos e de cobertura.

Conjunto	Número de pontos	Número de *clusters*	Total de arestas particionadas no grafo de conflitos	Total de arestas particionadas no grafo de cobertura	Total de restrições relaxadas pelo modelo $\overline{PPLAMC}$	Total de restrições relaxadas pelo modelo PPLAMC
C1	30	2	108	155	133	24
		4	224	261	253	30
		8	332	307	362	30
C2	324	5	361	1034	483	114
		10	742	2238	925	238
		20	1469	4186	1752	311
		30	2123	4364	2434	314
C3	818	5	11411	34457	11914	487
		10	21468	57579	22186	716
		20	36083	71157	36879	819
		30	42812	76283	43629	818
		Média	10648,5	22911,0	10995,5	354,6

Considerando o particionamento do grafo de cobertura $G = (V,E)$ em K partes com vértices $V_1, V_2, ..., V_K$, os sub-grafos $G_k = G[V_k]$ podem ser definidos, bem como suas arestas $E_k = E(G_k)$. Com isso, as arestas que conectam os sub-grafos, ou seja, as arestas particionadas, podem ser definidas como $\hat{E} = E \setminus \bigcup_{k=1}^{K} E_k$. Desse modo, após a fase de particionamento, a formulação do PPLAMC pode ser novamente escrita:

$$v(PPLAMC) = Max \sum_{k=1}^{K} \mathbf{a}^k \mathbf{x}^k \tag{4.32}$$

Sujeito a

$$\begin{bmatrix} A_1 & A_2 & ... & A_K \\ B_1 & 0 & 0 & 0 \\ 0 & B_2 & 0 & 0 \\ ... & ... & ... & ... \\ 0 & 0 & 0 & B_K \end{bmatrix} \begin{pmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}^1 \\ \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}^2 \\ \vdots \\ \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}^K \end{pmatrix} \approx \mathbf{b} \quad e \quad \mathbf{x}^k, \mathbf{y}^k \in B^{|V_k|} \quad \forall k \in \{1, .., K\} \tag{4.33}$$

Em que:

- $\mathbf{a}^k$ é um vetor de população associado com os vértices $v \in V_k$;
- A_k é uma matriz de coeficientes das restrições definidas em (4.3), associadas com o conjunto de vértices V_k e com o conjunto de arestas $\hat{E}$, e da restrição definida em (4.6);
- B_k é uma matriz de coeficientes das restrições associadas com o conjunto de vértices V_k definidas em (4.2), (4.4), (4.5), e das restrições definidas em (4.3) associadas às arestas de E_k;
- $\approx$ representa o operador relacional $=$ ou $\leq$ dependendo das respectivas restrições; e
- $\mathbf{b}$ é um vetor com os termos do lado direito das restrições.

Agora, relaxando no sentido lagrangeano as restrições de acoplamento dos blocos, ou seja, o conjunto de restrições definido pelas matrizes A_k em (4.33), cada subproblema k pode ser assim escrito:

$$v(PPLAMC)_k^{LC} = Max \left\{ \left(\mathbf{a}^k - \mathbf{A}_k^T \mu\right) \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}^k : \mathbf{x}^k, \mathbf{y}^k \in C_k \right\} \tag{4.34}$$

Em que μ é um vetor de multiplicadores lagrangeanos relacionados às restrições definidas pelas matrizes A_k em (4.33), e C_k representa o conjunto de restrições embutido no *cluster* k, ou seja, o bloco k das restrições definidas pelas matrizes B_k em (4.33).

Se μ é um vetor de dimensão Q, em que $\mu_q \in \Re^+ \quad \forall q = 1, ..., Q-1$, $\mu_Q \in \Re$ e Q é o número de restrições de acoplamentos definidas em (4.33), o limitante lagrangeano pode ser assim escrito:

$$v\left(PPLAMC^{LC}\right) = \sum_{k=1}^{K} \left\{v(PPLAMC)_k^{LC}\right\} + \sum_{q=1}^{Q-1} \mu_q + p\mu_Q \tag{4.35}$$

A decomposição do problema por meio de *clusters* permite a utilização da decomposição de Dantzig-Wolfe e, conseqüentemente, pode-se aplicar um algoritmo de geração de colunas ao PPLAMC, que será apresentado a seguir.

4.4 Decomposição Dantzig-Wolfe e o Algoritmo de Geração de Colunas para o PPLAMC

Conforme discutido no Capítulo 2, a formulação mostrada em ((4.32)-(4.33)) pode ser explorada em um Algoritmo de Geração de Colunas. Assim, aplicando-se a decomposição Dantzig-Wolfe (DW) no problema ((4.32)-(4.33)) tem-se o seguinte Problema Mestre Restrito:

$$v(PPLAMC_{DW})_{LP} = Max \sum_{k=1}^{K} \sum_{j \in J_k} \lambda_{jk} \left(\mathbf{a}^k \tilde{\mathbf{x}}^{jk}\right) \tag{4.36}$$

Sujeito a

$$\sum_{k=1}^{K} \sum_{j \in J_k} \lambda_{jk} \left(\mathbf{A}_k \tilde{\mathbf{w}}^{jk}\right) \approx \mathbf{c} \tag{4.37}$$

$$\sum_{j \in J_k} \lambda_{jk} = 1 \quad \forall k \in \{1, ..., K\} \tag{4.38}$$

$$\lambda_{jk} \geq 0 \quad \forall k \in \{1, ..., K\},\ j \in J_k \tag{4.39}$$

Em que:

- J_k : Conjunto de pontos extremos do *cluster* k;

- $\tilde{\mathbf{w}}^{jk} = \begin{bmatrix} \tilde{\mathbf{x}} \\ \tilde{\mathbf{y}} \end{bmatrix}^{jk}$: Vetor que define pontos extremos $j \in J_k$;

- $\mathbf{a}^k$, com $|V_k|$: Vetor de população associado com vértices $v \in V_k$; e

- λ_{jk} : Variáveis de decisão que correspondem a pontos extremos $j \in J_k$.

- $\mathbf{c} \subset \mathbf{b}$ representa o lado direito das restrições relacionadas com $\mathbf{A}$.

O subproblema k, k $\in \{1, ..., \text{K}\}$, tem a forma da expressão mostrada em (4.34), com o vetor de multiplicadores lagrangeanos substituídos pelo vetor de variáveis duais de programação linear α associadas com as restrições (4.37):

$$v(PPLAMC)_k^{GC} = Max \left\{ \left(\mathbf{a}^k - \mathbf{A}_k^T \alpha \right) \mathbf{w}^k : \mathbf{w}^k \in C_k \right\} \tag{4.40}$$

Em que $\mathbf{w}^k = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}^k$ e C_k são as restrições associadas ao subproblema k *(Cluster k)*.

Considerando o PMR da decomposição descrita em ((4.36)-(4.39)), uma nova coluna gerada pelo subproblema k é uma coluna a ser inserida no PMR se o seu custo reduzido for positivo, isto é, $v(PPLAMC)_k^{GC} - \beta_k > 0$, em que β_k é a variável dual associada com a k^{esima} restrição de convexidade de (4.38). Com isso, o PMR coordena as soluções dos problemas por meio de suas variáveis duais, buscando uma solução para o problema original.

Considerando que α é um vetor com Q posições ($\alpha \in R_+^Q$), em que $\alpha_q \in \Re^+, q = 1..Q-1, \alpha_Q \in \Re$ e Q é a quantidade de restrições de acoplamento, sendo a posição Q de α correspondente à restrição (4.6), a decomposição apresentada acima fornece um modo alternativo de se obter a relaxação LagClus para o PPLAMC ($LC_\alpha^{GC} PPLAMC$):

$$[h] v \left(LC_\alpha^{GC} PPLAMC \right) = \sum_{k=1}^{K} \left\{ v(PPLAMC)_k^{GC} \right\} + \sum_{q=1}^{Q-1} \alpha_q + p\alpha_Q \tag{4.41}$$

O conjunto inicial de colunas é composto apenas por soluções viáveis para o PPLAMC. O algoritmo mostrado na Figura 4.6 gera uma quantidade pré-definida dessas soluções.

```
Rotina Inserir_Solucão_Viável_RMP (Num_Solutions)
//Seja Num_Solutions o número de soluções viáveis iniciais
  Para i ← 1 até Num_Solutions faça
    Escolha p localizações aleatoriamente;
    Para i ← 1 até |I| Faça
      Alocar o ponto de demanda i à facilidade (previamente escolhida)
      que esteja dentro de sua área de cobertura, que tenha a menor
      valor de demanda acumulada e que não exceda a restrição de
      capacidade (4.4) ou (4.5).
      Aplique o algoritmo location-allocation proposto por Pereira e
      Lorena (2002) para melhorar a solução.
    Fim-para
    Insira a solução factível no PMR.
  Fim-Para
Fim Rotina
```

Figura 4.6 - Algoritmo para inserir soluções viáveis ao PMR.

Cada solução viável gerada pelo algoritmo acima representa K colunas no PMR, uma para cada *cluster*. A Figura 4.7 mostra o diagrama com os fluxos e os passos principais do método de geração de colunas usado neste trabalho. A área sombreada representa a repetição necessária para se produzir melhores colunas.

Os critérios de parada para o processo de geração de colunas são: nenhuma das novas colunas geradas tem custo reduzido positivo ou $v\left(PMR\right) \geq v\left(LC_{\alpha}^{GC}PPLAMC\right)$. Este último critério é possível, dado que o valor da LagClus é um limitante superior para o processo de geração de colunas e o valor do PMR é também um limitante superior quando esse teste é satisfeito. Com isso, o processo pode ser finalizado, mesmo tendo ainda colunas a acrescentar.

Para este problema não foi necessário remover colunas improdutivas, pois como será mostrado mais adiante, poucas colunas foram inseridas durante a aplicação do algoritmo.

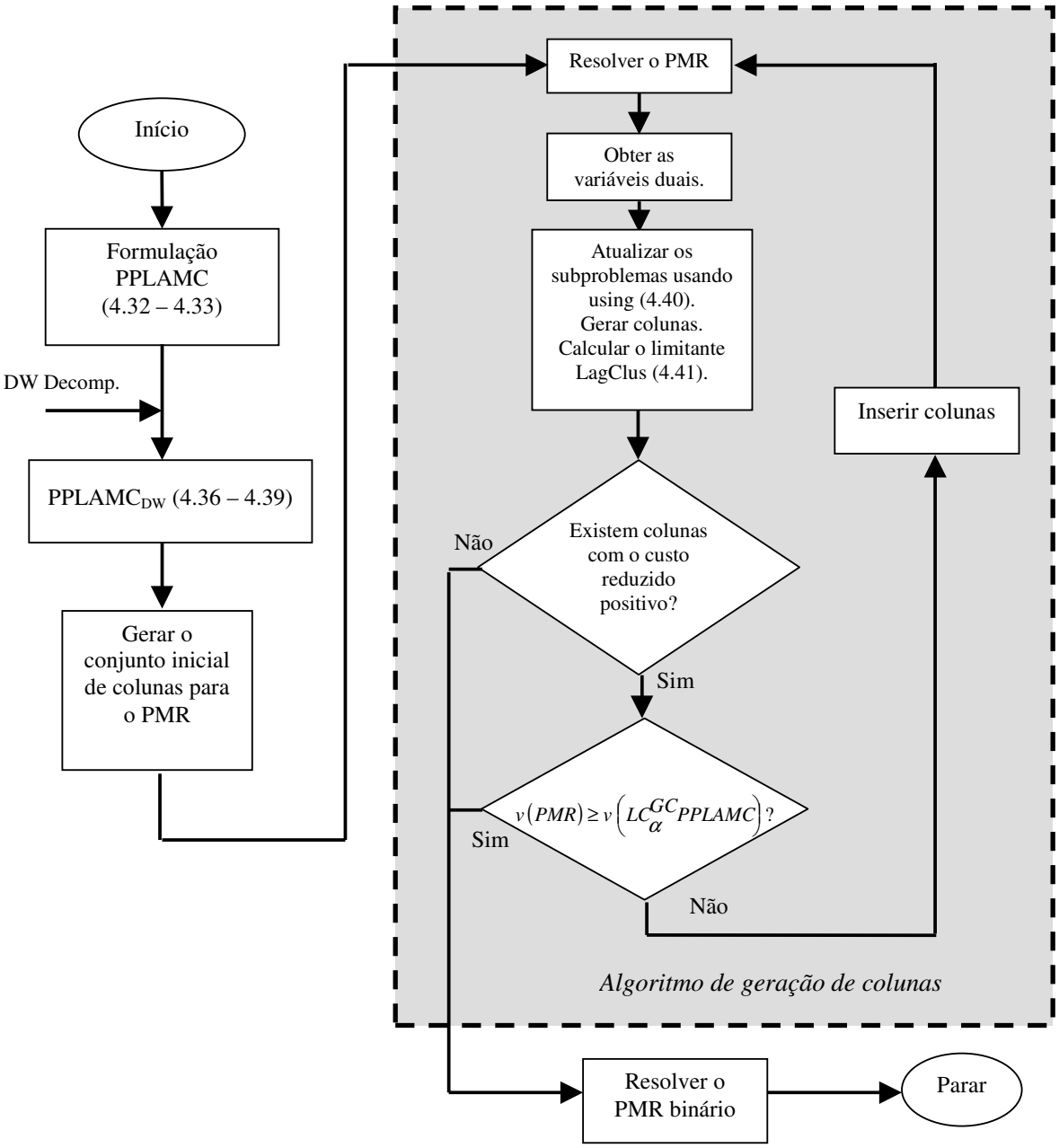


Figura 4.7 - Diagrama dos passos usados na abordagem da geração de colunas.

Fonte: adaptado de Ribeiro (2007).

4.5 Algoritmo Genético Aplicado ao PPLAMC

O Algoritmo Genético Construtivo (AGC) foi descrito no Capítulo 2 e, como foi dito, trabalha com uma população inicial composta por estruturas e esquemas. O símbolo # é chamado de curinga e representa um ponto ainda não definido para o problema. Neste trabalho, esse símbolo receberá o valor 1, tornando-se uma facilidade, ou o valor 2,

tornando-se um ponto de demanda que será atribuído a uma facilidade durante a evolução do processo. Desta forma, considera-se um esquema representado por uma cadeia que apresenta três tipos de informações:

- 1 : vértice que representa uma facilidade;
- 2 : vértice que representa um ponto de demanda que é atribuído a uma facilidade;
- # : vértice curinga, que se tornará uma facilidade ou um ponto de demanda a ser atribuído a uma facilidade durante o processo evolutivo.

A Figura 4.8 mostra um exemplo de uma cadeia para um problema com 10 pontos e 2 facilidades.

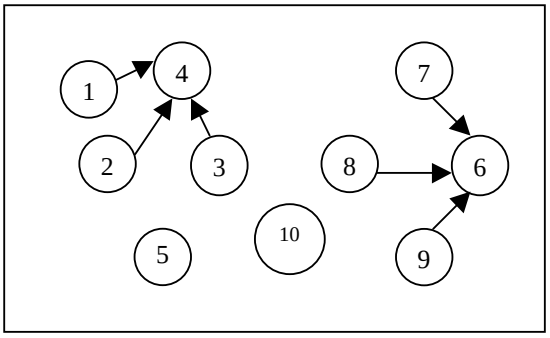


Figura 4.8 - Representação de S = (2221#1222#).

Fonte: adaptado de Furtado (1998).

A implementação do AGC para o PPLAMC considera a formulação ((4.1)-(4.7)) e a modelagem de cada indivíduo em um vetor, em que uma porcentagem das posições recebe aleatoriamente o valor 2 e exatamente p posições recebem o valor 1, definindo, assim, as p facilidades a serem abertas. As posições restantes recebem o valor #. Para os testes computacionais, 20% das posições receberam o valor 2. A Figura 4.9 mostra um possível indivíduo da população inicial para um problema com quatro centros e trinta nós.

2 # 1 # # # # 2 # # 1 # 2 # # 2 # 2 # # 1 # # # 1 # 2

Figura 4.9 - Possível indivíduo para um problema com quatro centros e trinta nós.

O operador de seleção base-guia escolhe duas estruturas para o cruzamento. A primeira, denominada base (S_{base}), é obtida dos 20% melhores indivíduos da população. A segunda, denominada guia (S_{guia}), é selecionada da população total. O operador de cruzamento compara essas duas estruturas em posições correspondentes, gerando uma nova estrutura filha (S_{nova}), com as seguintes regras baseadas em Furtado (1998):

- $S_{base} = \#$ e $S_{guia} = \#$ então $S_{nova} = \#$;
- $S_{base} = 1$ e $S_{guia} = 1$ então $S_{nova} = 1$;
- $S_{base} = 2$ e $S_{guia} = 2$ então $S_{nova} = 2$;
- $S_{base} = 1$ e $S_{guia} = \#$ então $S_{nova} = 1$;
- $S_{base} = 2$ e $S_{guia} = \#$ então $S_{nova} = 2$;
- $S_{base} = \#$ e $S_{guia} = 2$ então $S_{nova} = 2$;
- $S_{base} = \#$ ou 2 e $S_{guia} = 1$ então $S_{nova} = 1$ ou 2, escolhido aleatoriamente; e
- $S_{base} = 1$ e $S_{guia} = 2$ então $S_{nova} = 1$ ou 2, escolhido aleatoriamente.

Após a aplicação do operador de cruzamento, pode-se ter mais do que p valores *1* em S_{nova}. Essa condição não é relaxada e uma validação da nova estrutura é feita para se garantir exatamente p facilidades, inserindo ou eliminando aleatoriamente valores *1* nessa nova estrutura.

O operador de mutação faz uma troca de um centro com um vizinho que não tem a sua demanda atendida. Isso aumenta a chance de todos os pontos de demanda serem centros. O algoritmo do operador de mutação é mostrado na Figura 4.10.

```
Rotina Operador_Mutação
  Verificar se existem pontos de demanda que não são atendidos
  Se existirem
  Então
    Escolher um deles aleatoriamente e atribuir-lhe o valor 1 (defini-
      lo como facilidade)
    Escolher uma posição que contenha uma facilidade e atribuir-lhe o
      valor 2
  Fim-Se;
Fim Rotina
```

Figura 4.10 - Operador de mutação.

O tratamento das restrições (4.2), (4.3) e (4.4) ou (4.5) é baseado no *loop* interno do algoritmo da Figura 4.6. As p facilidades exigidas na restrição (4.6) são consideradas na própria definição do indivíduo, conforme exemplo na Figura 4.9. Definidas as localizações e as alocações, o valor da função objetivo é obtido de forma trivial.

4.6 *Clustering Search* Aplicado ao PPLAMC

Discute-se a seguir uma versão do CS para a resolução do PPLAMC. Uma solução é representada por uma estrutura computacional que contém os centros de serviços localizados, a alocação dos pontos de demanda e o valor da função objetivo. A Figura 4.11 mostra um exemplo de centros de serviços localizados em uma possível solução com $p = 5$.

3	8	12	19	25

Figura 4.11 - Centros de serviços localizados em uma possível solução com $p = 5$.

O componente *MH*, responsável por gerar soluções para o processo de agrupamento, foi implementado utilizando a metaheurística *Greedy Randomized Adaptive Search Procedure* - GRASP (FEO; RESENDE, 1995). O GRASP é basicamente composto por duas fases: uma de construção, na qual uma solução viável é gerada e uma fase de busca local, na qual a solução construída é melhorada.

A fase de construção do GRASP foi baseada na heurística proposta por Marianov e Serra (1998) e é mostrada na Figura 4.12. Existem duas diferenças da heurística original. Primeira, no Passo *5* é escolhido o centro com a mais alta taxa de chamada corrente. Segunda, após o Passo *5* é feita uma busca local, que no caso do PPLAMC já é aplicada na segunda fase do GRASP.

A fase de busca local do GRASP usa a heurística de localização-alocação proposta por Lorena e Pereira (2002). As soluções obtidas na fase de construção podem ser melhoradas com essa nova heurística, buscando-se um novo centro de serviços dentro de cada área de cobertura, substituindo-se o centro de serviços corrente por um vértice da mesma área. Após essa troca, o atendimento dos pontos de demanda deve ser recalculado, pois a alocação e a configuração de cobertura desses pontos podem ser modificadas.

Heurística Construtiva
Passo 1:
Faça uma lista (D_j) das posições candidatas a facilidades, ordenadas decrescentemente por taxa de chamada.
Para cada localização candidata, calcule o lado direito da equação (4.4) ou (4.5), dependendo do modelo aplicado.
Para cada localização candidata j, faça uma lista (D_{ji}) de todos os pontos de demanda i dentro da distância padrão, ordenados por distâncias crescentes à localização candidata.
Passo 2:
Para cada localização candidata, faça a taxa total de chamada corrente igual a zero.
Passo 3:
Começando com a primeira localização j da lista D_j, some à sua taxa de chamada, a taxa f_i do primeiro ponto da lista D_{ji}. Então, adicione a taxa f_I do segundo ponto, e assim por diante, até que qualquer extra valor adicionado exceda o limite de chamadas de j. Aloque temporariamente todos esses pontos de demanda ao centro hipotético no nó j.
Passo 4:
Repita o Passo 3 para todas as localizações candidatas de D_j. Deve-se observar que um mesmo ponto de demanda pode estar temporariamente alocado a vários candidatos a centro.
Passo 5:
Localize um centro de serviços em um ponto escolhido aleatoriamente dentro dos pontos com mais alta taxa de chamada corrente. Retire todos os pontos de demanda alocados a este das lista D_{ji} e os aloque definitivamente a este.
Passo 6:
Repita os Passos 2 a 5 até que todas as facilidades estejam localizadas.
Fim-Heurística

Figura 4.12 - Heurística construtiva do GRASP.

O componente AI realiza um agrupamento iterativo de cada solução obtida pelo GRASP e trabalha como um classificador, mantendo no sistema somente informações relevantes e direcionando a busca nas áreas promissoras. Um número máximo de regiões é definido *a priori*. A i^{esima} região tem o seu próprio centróide C_i e um raio r que é comum às demais.

A métrica de distância, que calcula a similaridade entre o centróide das regiões e a solução recebida do componente MH, é definida neste trabalho como o número de diferentes centros de serviços localizados entre a solução GRASP e a solução do centróide da região. Sendo assim, quanto maior o número de centros de serviços diferentes entre duas soluções, maior será a distância entre elas e menor será a similaridade. A solução GRASP será, então, atribuída à região com menor distância métrica.

No processo de assimilação é utilizado o método *path-relinking* (GLOVER, 1996), que realiza movimentos exploratórios na trajetória que interconecta a solução gerada pelo GRASP e a do centróide do agrupamento mais similar (C_i). Assim sendo, o próprio

processo de assimilação já é uma forma de busca local dentro do agrupamento, pois o centróide vai ser deslocado para a melhor solução avaliada nessa trajetória, caso ela seja melhor que o centróide corrente. A Figura 4.13 apresenta um exemplo do *path-relinking* aplicado ao PPLAMC, em que no centróide do agrupamento está uma solução com os centros localizados nos pontos 3, 8, 12, 19 e 25, e a nova solução tem as localizações em 3, 15, 21, 7, e 30. Para essas soluções a distância métrica é 4.

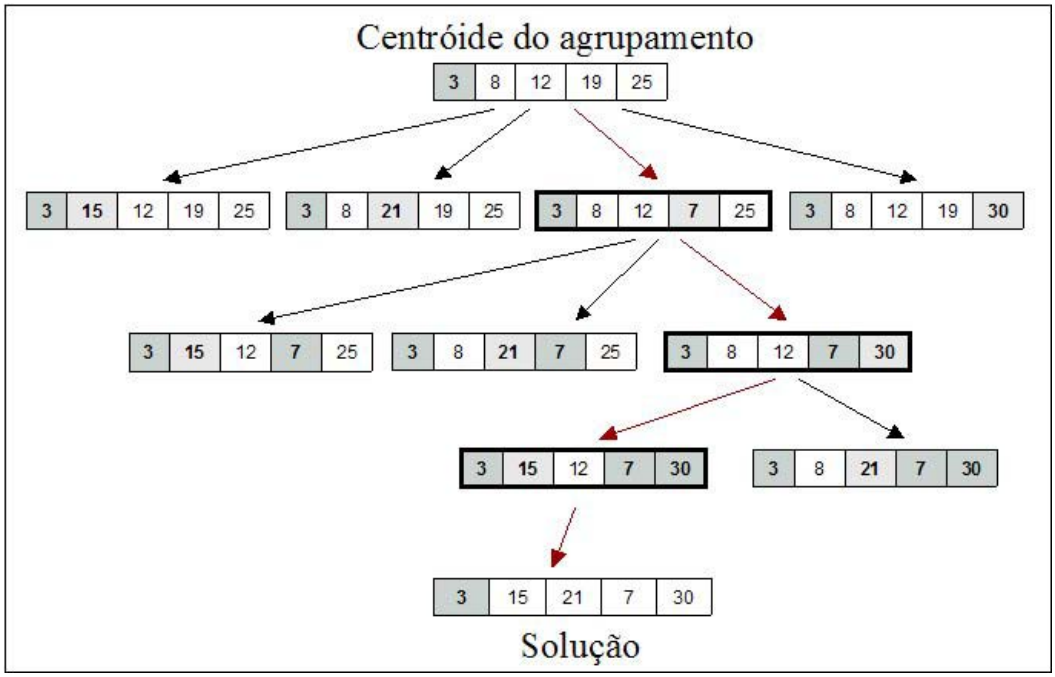


Figura 4.13 - *Path-relinking* aplicado ao PPLAMC.

O componente AA verifica se o agrupamento já pode ser considerado promissor, isto é, quando a quantidade de soluções a ele designadas ultrapassa um certo valor χ_i, definido *a priori*. Este componente é executado toda vez que uma solução for atribuída a um agrupamento.

O componente AO é ativado sempre que AA descobre uma região promissora. A sua implementação baseia-se também na heurística de localização-alocação proposta por Lorena e Pereira (2002), utilizada na fase de busca local do GRASP, modificando-se a maneira de selecionar um novo centro de serviços. Em vez de trocar um centro corrente por um vértice da sua área de cobertura, troca-se o centro corrente por um vértice não alocado, escolhido aleatoriamente, modificando-se a solução de alocação e recalculando-se a cobertura. Esse processo permite buscar boas soluções em diferentes partes do espaço de busca em relação ao GRASP.

O pseudo-código do CS é apresentado na Figura 4.14.

```
Procedimento CS
   Para (número de iterações não satisfeito) faça
// componente MH
      s = ∅
      Enquanto (solução não construída) Faça
        Produzir a Lista de Candidatos (LC)
        e = selecionar elemento aleatoriamente dentro
            dos 20% melhores.
        s = s ∪ {e}
        Atualizar lista de Candidatos(C)
      Fim-Enquanto
      BuscaLocal(s)
      Aplicar o componente AI (s)
      Aplicar o componente AA (região ativa)
      Se (região ativa for promissora) Então
        Aplicar o componente AO
   Fim-Para
Fim-Procedimento CS
```

Figura 4.14 - Pseudo-código do CS aplicado ao PPLAMC.

4.7 Resultados Computacionais

Para os experimentos computacionais foram utilizadas redes de 30, 324 e 818 nós. A primeira foi proposta por Marianov e Serra (1998) e encontra-se mostrada na Tabela 4.3. As demais foram criadas a partir de uma base de dados geográficos da cidade de São José dos Campos-SP, acrescida da população fictícia em cada ponto de demanda. Estas redes estão disponíveis em http://www.lac.inpe.br/~lorena/instancias.html. Várias instâncias foram montadas, variando-se os parâmetros p, b, ϕ, φ e τ.

Para a implementação do problema de máxima cobertura, os centros de serviços foram considerados postos de saúde, com um médico em cada centro. Cada ponto de demanda é um potencial centro de atendimento e as distâncias são euclidianas.

Para a rede com 30 vértices considerou-se: raio de cobertura igual a 1,5 milhas, o tempo médio de atendimento igual a 20 min e a taxa de chamada diária igual a 0,015 vezes a população do ponto de demanda para as restrições do comprimento da fila e 0,006 vezes a população do ponto de demanda para as restrições do tempo de espera, todos definidos em Marianov e Serra (1998).

Para a rede com 324 vértices considerou-se: raio de cobertura igual a 250m, e para os de

818, 750m. Para as duas redes foi considerado o tempo médio de atendimento igual a 15 min e a taxa de chamada igual a 0,01 x população do ponto de demanda, tanto para as restrições do comprimento da fila, como para as restrições do tempo de espera.

Tabela 4.3 - Rede de 30 nós proposta por Marianov e Serra (1998)

Nó	X	Y	População	Nó	X	Y	População
1	3.2	3.1	710	16	3.0	5.1	110
2	2.9	3.2	620	17	1.9	4.7	100
3	2.7	3.6	560	18	1.7	3.3	100
4	2.9	2.9	390	19	2.2	4.0	90
5	3.2	2.9	350	20	2.5	1.4	90
6	2.6	2.5	210	21	2.9	1.2	90
7	2.4	3.3	200	22	2.4	4.8	80
8	3.0	3.5	190	23	1.7	4.2	80
9	2.9	2.7	170	24	6.0	2.6	80
10	2.9	2.1	170	25	1.9	2.1	80
11	3.3	2.8	160	26	1.0	3.2	70
12	1.7	5.3	150	27	3.4	5.6	60
13	3.4	3.0	140	28	1.2	4.7	60
14	2.5	6.0	120	29	1.9	3.8	60
15	2.1	2.8	120	30	2.7	4.1	60

Os nomes dos problemas foram codificados do seguinte modo: quantidade de pontos, quantidade de centros de serviços, tipo de restrição (0 para a quantidade de pessoas na fila e 1 para o tempo de espera), quantidade de pessoas na fila ou tempo de espera, e probabilidade. Exemplo: 324_20_0_2_95 significa 324 pontos, 20 centros, restrição para a quantidade de pessoas na fila, máximo de duas pessoas na fila com probabilidade de, no mínimo, 95%.

Os tempos nas tabelas estão em segundos e foram obtidos em um computador Pentium IV 3GHz com 1GB RAM. Todos os experimentos foram limitados em três horas de processamento (10800s). O particionamento do grafo de cobertura foi realizado pela heurística METIS (KARYPIS; KUMAR, 1998a), que busca balancear o número de vértices nos *clusters*. Os subproblemas da LagClus, a relaxação de programação linear e o PMR foram resolvidos usando o CPLEX 10. Entretanto, mesmo usando esse *solver*, não foi possível resolver os subproblemas da LagClus e os subproblemas geradores de colunas para o PMR em três horas de processamento para algumas instâncias do PPLAMC, ou seja, não se conseguiu executar um passo do algoritmo de subgradientes ou uma repetição do algoritmo de geração de colunas. Para evitar esse tempo excessivo, utilizou-se um algoritmo que limita o tempo de cada subproblema, permitindo assim a aplicação

da LagClus e do Algoritmo de Geração de Colunas, conforme descrito na Figura 4.15. Considerando a LagClus, se algum subproblema não for resolvido de forma ótima (variável limitante_válido = 0), mesmo tendo uma solução viável inteira para esse *cluster*, o valor do dual lagrangeano não pode ser considerado um limitante, porém a solução inteira é utilizada para compor a solução relaxada do PPLAMC. Considerando o Algoritmo de Geração de Colunas, mesmo que algum subproblema não seja resolvido de forma ótima, a solução inteira encontrada para esse *cluster* pode ser inserida no PMR.

As Tabelas 4.5, 4.6 e 4.7 comparam os métodos utilizados em 26 instâncias da rede de 30 nós, as Tabelas 4.8, 4.9 e 4.10 comparam os métodos aplicados em 24 instâncias da rede de 324 vértices e as Tabelas 4.11, 4.12 e 4.13 comparam os métodos aplicados em 24 instâncias da rede de 818 vértices.

```
Procedimento Limitar_Tempo_Subproblema
  Configurar o tempo_máximo para execução do subproblema
    no CPLEX
  Definir um valor de acréscimo ao tempo máximo (delta)
  Fazer limitante_válido = 1
  Fazer limitante = 0
  Para (cada subproblema) Faça
     Resolver subproblema no CPLEX
     Enquanto (solução viável não encontrada) Faça
        tempo_máximo = tempo_máximo + delta
        Continuar resolvendo o subproblema no CPLEX
     Fim-Enquanto
     Se (solução não é ótima)
        limitante_válido = 0
     Fazer limitante = limitante + solução
  Fim-Para
  Retornar variáveis limitante_valido e limitante
Fim-Procedimento Limitar_Tempo_Subproblema
```

Figura 4.15 - Algoritmo usado para resolver os subproblemas em tempo limitado.

4.7.1 Resultados do PPLAMC obtidos via CPLEX, Heurística e Metaheurísticas

As Tabelas 4.5, 4.8 e 4.11 mostram os resultados da aplicação do CPLEX, heurística e metaheurísticas ao PPLAMC e são divididas em seis grupos: o nome da instância, os resultados do CPLEX, os resultados da heurística proposta por Marianov e Serra (1998), os resultados do AGC, os resultados do GRASP (componente *MH* do *CS*) e os resultados

do *CS*. O valor em negrito mostra a melhor solução encontrada para a instância.

Nas colunas referentes ao CPLEX são fornecidos: a solução inteira viável (coluna Solução) e o *gap* obtido do CPLEX 10.0 (coluna *Gap C*), definido como o desvio percentual entre os limitantes superior (UB) e inferior (LB), calculado como: $\frac{(UB-LB)}{LB} * 100$. Para a rede de 30 nós, o CPLEX não encontrou a solução ótima em três instâncias (cerca de 12% dos casos estudados). Considerando a rede de 324 nós, os problemas marcados com $*$ na coluna *Gap C* tiveram a execução interrompida por falta de memória. Além disso, o CPLEX não encontrou o valor ótimo em oito instâncias. Para a rede de 818 vértices o CPLEX não encontrou o valor ótimo em cerca de 46% das instâncias testadas, sendo que em quatro, os *gaps* ficaram acima de 100%.

Os resultados da heurística proposta por Marianov e Serra (1998) refletem as soluções de cada instância mostradas em três colunas: o valor encontrado (coluna Solução), o tempo de execução (coluna Tempo) e o desvio (coluna *Gap H*), calculado como 100*(Solução CPLEX - Solução da heurística)/(Solução CPLEX).

Os resultados do *AGC* e do *CS* refletem cinqüenta execuções de cada instância e são mostrados em quatro colunas: o melhor valor encontrado (coluna Melhor), o valor médio da função objetivo (coluna Média), o tempo médio (colunas Tempo) e o desvio (coluna *Gap G* e *Gap CS*, respectivamente) que define, em porcentagem, o erro relativo entre os valores das colunas Solução CPLEX e Melhor, calculado por 100*(solução CPLEX - Melhor)/(Solução CPLEX). O valor de χ foi considerado igual a 6, ajustado após várias execuções, fornecendo bons resultados para o *CS*. Valor negativo de *gap* indica que a melhor solução do *AGC* ou do *CS* foi melhor do que a do CPLEX.

Os resultados do GRASP refletem cinqüenta execuções de cada instância e são mostrados em três colunas: o melhor valor encontrado (coluna Melhor), o valor médio da função objetivo (coluna Média) e o tempo médio (coluna Tempo). O GRASP (componente MH do *CS*) encontrou piores valores para a função objetivo que o *CS* em cerca de 8% das instâncias, considerando a rede de 30 nós; em 75% das instâncias, considerando a rede de 324 nós, e em cerca de 92%, para a rede de 818 nós. Isto mostra que os demais componentes do *CS* contribuem para melhorar os valores das soluções, porém, com um custo computacional maior.

Para todas as redes, de modo geral, as soluções do CS foram melhores que as da heurística do Marianov e Serra (1998) e das demais metaheurísticas. Para a rede de 30 nós, o CS forneceu o mesmo valor encontrado pelo CPLEX em cerca de 65% das instâncias, considerando os valores médios, e em cerca de 88%, considerando a melhor solução, com

melhor tempo médio. Para a rede de 324 o CS foi melhor que CPLEX apenas no tempo médio. Para a rede de 818 nós e considerando a melhor solução, o CS encontrou valores iguais ao CPLEX em 58% das instâncias e valores melhores em cerca de 33%, com melhor tempo médio.

Os resultados do AGC foram obtidos usando-se os valores dos parâmetros mostrados na Tabela 4.4, ajustados após várias execuções do programa.

Tabela 4.4 - Parâmetros usados no AGC

Parâmetro	Instâncias com 30 e 324 pontos
GMax	1,1* soma da população de todos os pontos de demanda
d	0,1
e	0,001
l	0,0001
Cruzamentos por geração	30
Probabilidade de mutação	0,20
População inicial	300
Quantidade máxima de gerações	300

4.7.2 Resultados do PPLAMC Obtidos com Relaxações

Um algoritmo de subgradientes adaptado para a LagClus e para a relaxação lagrangeana foi implementado conforme Parker e Rardin (1988), Narciso e Lorena (1999) e Wolsey (1998). O controle utilizado para o passo (parâmetro π) foi o proposto por Held e Karp (1970), que se inicia com o valor 2 e é dividido à metade se durante 30 iterações consecutivas o limitante superior não decrescer. As condições de parada foram:

- $\pi \leq 0,005$;
- (limite superior - limite inferior) < 1;
- Quadrado da norma euclidiana do subgradiente $= 0$.

As Tabelas 4.6, 4.9 e 4.12 mostram os resultados das relaxações aplicadas ao PPLAMC e são divididas em quatro grupos: o nome da instância, os resultados da relaxação de programação linear, os resultados da relaxação lagrangeana e os resultados da LagClus. A coluna *Gap PL* nessas tabelas foi calculada da seguinte maneira: $Gap\ PL(\%) = \frac{(\text{Limite Superior} - \text{Solucao CPLEX})}{\text{Solucao CPLEX}} \times 100$. As colunas *Gap Lag* e *Gap LC* nessas tabelas foram calculadas da seguinte maneira: $Gap(\%) = \frac{(\text{Limite Superior} - \text{Limite Inferior})}{\text{Limite Inferior}} \times 100$. Os limites inferiores para a relaxação lagrangeana foram obtidos com o algoritmo MTHG, conforme

descrito em 4.2. Para a LagClus, foram utilizadas as melhores soluções encontradas na literatura disponíveis nos trabalhos de Corrêa e Lorena (2006), Corrêa et al. (2007) e Corrêa et al. (2008).

Em todas essas tabelas, nota-se que os limitantes de programação linear são altos se comparados com os do CPLEX, da relaxação lagrangeana e os da LagClus. As linhas em negrito indicam soluções ótimas encontradas pelos métodos. A Lag Clus foi aplicada às instâncias do PPLAMC modeladas por grafos de cobertura. Esses grafos foram particionados em 2, 10 e 20 *clusters* para, respectivamente, as instâncias com 30, 324 e 818 nós.

Para as instâncias mostradas na Tabela 4.6, a relaxação de programação linear apresenta um tempo computacional baixo, porém o *gap* médio é o maior, com 32,3%. Analisando agora as demais relaxações, percebe-se que a relaxação lagrangeana encontra várias soluções ótimas e *gap* médio 0,92%, enquanto a LagClus encontra apenas uma, porém com o menor *gap* médio, de 0,77%.

Para as instâncias mostradas na Tabela 4.9, com 324 nós, a relaxação de programação linear também apresenta um tempo computacional baixo, ainda com o maior *gap* médio, de 246,7%. A relaxação lagrangeana encontra várias soluções ótimas com *gap* médio de 0,008%, menor que o encontrado com a LagClus, com 0,07%. A importante vantagem da LagClus sobre as demais relaxações está nas instâncias de grande porte, como pode ser observado a seguir.

A Tabela 4.12 apresenta os resultados para as instâncias de grande porte com 818 pontos. O *gap* médio da relaxação de programação linear é significativo, maior que 800%. Também problemática para estas instâncias é a relaxação lagrangeana, que não conseguiu executar a primeira iteração do algoritmo de subgradientes em três horas de processamento. Conseqüentemente, nenhum *gap* pôde ser calculado. Analisando agora os resultados da LagClus, percebe-se que o tempo computacional médio utilizado por ela é comparável ao do CPLEX, com média em torno de 7500s. Entretanto, a LagClus apresenta um *gap* médio menor, em torno de 28%, contra 73%, do CPLEX.

Desta forma, com o objetivo de investigar apenas os limitantes duais da LagClus foram utilizados limitantes inferiores fixos, com as melhores soluções encontradas na literatura (CORRÊA; LORENA, 2006; CORRÊA et al., 2007) e reportadas nas Tabelas 4.5, 4.8 e 4.11.

A abordagem da geração de colunas mostrou-se mais interessante que as relaxações, como será apresentado a seguir.

4.7.3 Resultados do PPLAMC Obtidos com o Algoritmo de Geração de Colunas

As Tabelas 4.7, 4.10 e 4.13 mostram os resultados da aplicação do algoritmo de geração de colunas ao PPLAMC, que são apresentados em dez informações para cada instância:

- Instância - nome da instância;
- Número de *clusters* - quantidade de *clusters* usados para o particionamento do grafo de cobertura.
- Número inicial de colunas - quantidade de colunas considerado no PMR inicial;
- PMR inicial - valor inicial obtido do PMR;
- Número final de colunas - quantidade final de colunas do PMR;
- PMR Final - valor final obtido do PMR;
- Tempo GC - Tempo de processamento da geração de colunas;
- Solução IP - valor obtido com o PMR resolvido usando variáveis inteiras;
- Tempo IP - Tempo de processamento do PMR com variáveis inteiras.
- Gap GC - Calculado por $\frac{(\text{PMR final} - \text{solucao inteira do PMR})}{\text{solucao inteira do PMR}} * 100$. *Gap* GC igual a zero significa que o valor ótimo foi obtido (valores em negrito).

Para a rede de 30 nós, na Tabela 4.7, o tempo de execução do algoritmo de geração de colunas foi competitivo comparado ao CPLEX, com bons valores de solução, em que também foram provados dez valores ótimos.

Para a rede de 324 nós, dez instâncias foram difíceis para o CPLEX, ou por problemas de memória ou por não encontrar o valor ótimo em três horas de processamento. Como mostrado na Tabela 4.10, o algoritmo de geração de colunas conseguiu obter quatro valores ótimos entre essas dez instâncias. O tempo de execução da abordagem da geração de colunas foi bastante competitivo em relação ao CPLEX, com bons valores de solução. As instâncias marcadas com um * definem uma parada de execução por limite de tempo. Nesses casos, o valor do PMR final não pode ser usado como limitante para provar a otimalidade da solução inteira. Portanto, seus *gaps* são estimados. Os dezoito valores ótimos encontrados são mostrados em negrito.

Os critérios de parada para o processo de geração de colunas são: nenhuma das novas colunas geradas tem custo reduzido positivo ou $v\left(LC_{\alpha}^{GC}PPLAMC\right) - v\left(RMP\right) < 1$, dado que, para as instâncias testadas, os coeficientes da função objetivo definida em (4.1) são valores inteiros. Além disso, por serem inteiros esses coeficientes, uma solução dada pelo PMR com variáveis binárias pode ser considerada ótima se a diferença entre os valores do PMR binário e do PMR final for menor que 1, uma vez que o limitante superior do algoritmo de geração de colunas é equivalente ao da LagClus, conforme (4.41).

Para a rede de 818 nós, o tempo de processamento da geração de colunas foi bastante competitivo em relação ao CPLEX, com melhores soluções em cerca de 42% das instâncias. O CPLEX não encontrou o valor ótimo em cerca de 46% das instâncias. Para todas as instâncias que o CPLEX encontrou solução ótima, o algoritmo de geração de colunas também encontrou, com melhores tempos computacionais. Além dessas, a geração de colunas provou a otimalidade de outras nove não provadas pelo CPLEX em três horas de processamento. Além disso, a média dos *gaps* para a abordagem de geração de colunas, cerca de 0,003%, é insignificante comparada às relaxações e ao CPLEX.

Um resultado interessante é que a geração de colunas insere um pequeno conjunto de novas colunas no PMR. São inseridas, em média, 30 colunas para a rede de 30 nós, 78 para a de 324 nós e 26 para a rede de 818. Este mesmo comportamento foi observado por Ribeiro e Lorena (2007), Ribeiro e Lorena (2008a).

A Figura 4.16 mostra o comportamento do Algoritmo de Geração de Colunas para uma instância específica. Deve-se observar que, no final, o limitante LagClus encontra o valor do PMR. Isto valida a equação (4.41) e indica que o PMR final é um limitante para o PPLAMC.

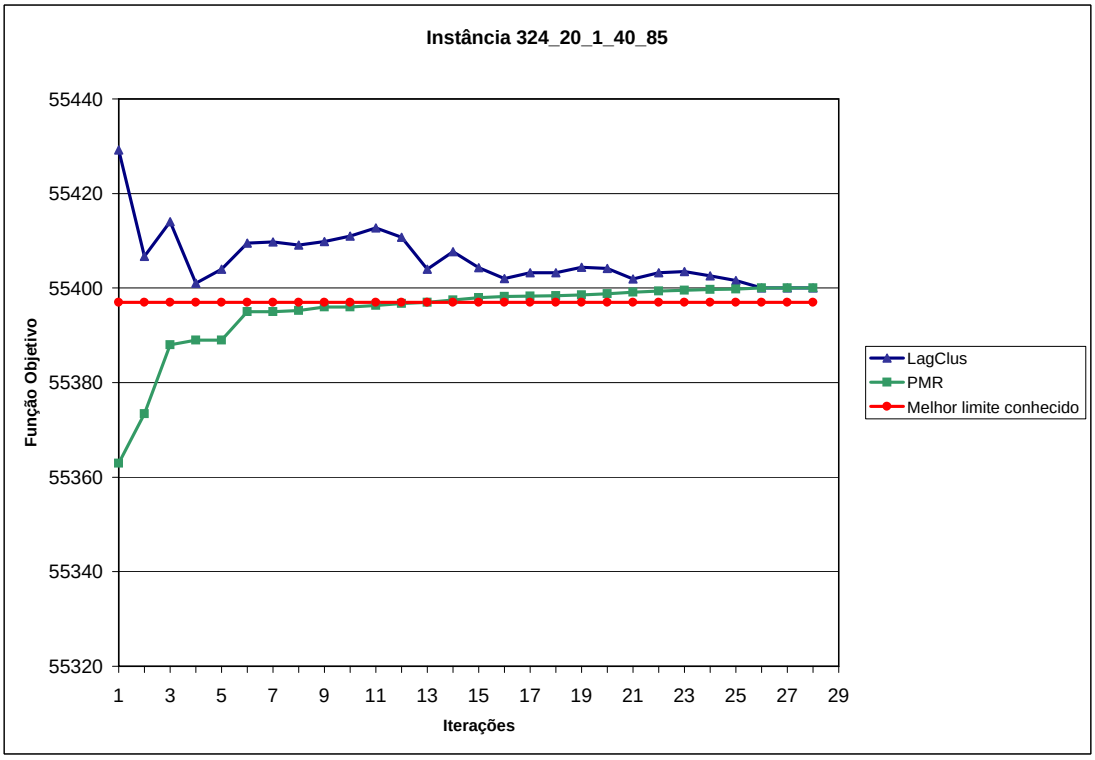


Figura 4.16 - Convergência da geração de colunas para a instância 324_20_1_40_85.

Tabela 4.5 - Resultados do CPLEX, heurística e metaheurísticas para a rede de 30 nós.

Instância	CPLEX			Heurística			AGC				GRASP			CS			
	Solução	Gap C	Tempo	Solução	Gap H	Tempo	Melhor	Média	Gap G	Tempo	Melhor	Média	Tempo	Melhor	Média	Gap CS	Tempo
30_2_0_0_85	**3700**	0	0,22	**3700**	0	0,000	**3700**	**3700**	0	0,31	**3700**	**3700**	0,007	**3700**	**3700**	0	0,009
30_3_0_0_85	**5390**	0	0,12	5210	3,34	0,000	**5390**	**5390**	0	0,51	**5390**	**5390**	0,019	**5390**	**5390**	0	0,025
30_2_0_1_85	**5100**	0	3,31	4630	9,22	0,000	5090	5090	0,20	0,36	5090	5090	0,016	5090	5090	0,20	0,018
30_3_0_1_85	**5390**	0	0,08	5210	3,34	0,000	**5390**	**5390**	0	0,48	**5390**	**5390**	0,021	**5390**	**5390**	0	0,024
30_2_0_2_85	**5210**	0	0,20	4780	8,25	0,000	**5210**	**5210**	0	0,38	**5210**	**5210**	0,015	**5210**	**5210**	0	0,016
30_3_0_2_85	**5390**	0	0,12	5210	3,34	0,000	**5390**	**5390**	0	0,47	**5390**	**5390**	0,022	**5390**	**5390**	0	0,023
30_5_0_0_95	**5330**	0,38	10800	5210	2,25	0,000	**5330**	5323	0	0,80	**5330**	5312,8	0,032	**5330**	5317,6	0	0,041
30_6_0_0_95	**5410**	1,11	10800	5390	0,37	0,000	**5410**	5392	0	0,84	5390	5390	0,029	**5410**	5391	0	0,037
30_3_0_1_95	**5270**	0	116,00	5080	3,61	0,000	5240	5240	0,57	0,48	5240	5239,8	0,022	5240	5240	0,57	0,024
30_4_0_1_95	**5390**	0	3,77	5260	2,41	0,000	**5390**	**5390**	0	0,54	**5390**	**5390**	0,022	**5390**	**5390**	0	0,026
30_2_0_2_95	**4520**	0	2,11	4470	1,11	0,000	**4520**	4513	0	0,30	**4520**	**4520**	0,013	**4520**	**4520**	0	0,015
30_3_0_2_95	**5390**	0	0,12	5230	2,97	0,000	**5390**	**5390**	0	0,50	**5390**	**5390**	0,023	**5390**	**5390**	0	0,027
30_4_1_48_90	**1920**	0	0,41	1890	1,56	0,000	**1920**	**1920**	0	0,66	**1920**	**1920**	0,011	**1920**	**1920**	0	0,014
30_5_1_48_90	**2400**	0	0,41	2280	5,00	0,000	2390	2390	0,42	0,74	**2400**	2398,8	0,013	**2400**	2398,8	0	0,019
30_3_1_49_90	**2160**	0	0,31	**2160**	0	0,000	**2160**	**2160**	0	0,53	**2160**	**2160**	0,007	**2160**	**2160**	0	0,009
30_4_1_49_90	**2880**	0	0,41	2870	0,35	0,000	**2880**	2877	0	0,59	**2880**	**2880**	0,008	**2880**	**2880**	0	0,013
30_5_1_50_90	**4700**	0	4,09	4670	0,64	0,000	**4700**	**4700**	0	0,73	**4700**	**4700**	0,018	**4700**	**4700**	0	0,028
30_6_1_50_90	**5390**	0	350,52	5060	6,12	0,000	**5390**	**5390**	0	0,97	**5390**	**5390**	0,027	**5390**	**5390**	0	0,038
30_5_1_40_85	**3050**	0	3,66	2910	4,59	0,000	3020	3001	0,98	0,74	**3050**	3033,2	0,016	**3050**	3033,2	0	0,021
30_6_1_40_85	**3610**	0	31,62	3480	3,60	0,000	**3610**	**3610**	0	0,91	**3610**	3606,8	0,021	**3610**	3608,8	0	0,031
30_7_1_40_85	**4060**	0	16,89	3860	4,93	0,000	**4060**	**4060**	0	1,03	**4060**	**4060**	0,028	**4060**	**4060**	0	0,039
30_6_1_41_85	**5330**	0,19	10800	5120	3,94	0,000	5300	5274	0,56	1,00	5270	5270	0,028	**5330**	5286,4	0	0,038
30_7_1_41_85	**5410**	0	43,56	5300	2,03	0,000	5390	5390	0,37	0,88	5390	5390	0,023	5390	5390	0,37	0,035
30_8_1_41_85	**5470**	0	0,05	5390	1,46	0,000	**5470**	**5470**	0	0,95	**5470**	**5470**	0,019	**5470**	**5470**	0	0,032
30_4_1_42_85	**4600**	0	0,75	4550	1,09	0,000	**4600**	**4600**	0	0,61	**4600**	**4600**	0,016	**4600**	**4600**	0	0,022
30_5_1_42_85	**5390**	0	6,50	5210	3,34	0,000	**5390**	**5390**	0	0,77	**5390**	**5390**	0,027	**5390**	**5390**	0	0,035
Média	**4533,1**	0,065	1268,6	4389,6	3,033	0,000	4528,1	4525	0,119	0,66	4527,7	4526,2	0,019	4530,8	4527,2	0,044	0,025

Tabela 4.6 - Resultados das relaxações para a rede de 30 nós.

Instância	Relaxação de Programação Linear			Relaxação Lagrangeana				LagClus			
	Limite superior	Gap PL (%)	Tempo (s)	Limite inferior	Limite superior	Gap Lag (%)	Tempo (s)	Limite inferior	Limite superior	Gap LC (%)	Tempo (s)
30_2_0_0_85	5310,9	43,54	0,03	**3700,0**	**3700,0**	**0,00**	**0,36**	**3700,0**	**3700,5**	**0,01**	**85,20**
30_3_0_0_85	5430,0	0,74	0,02	5390,0	5517,9	2,37	39,72	5390,0	5397,4	0,14	228,20
30_2_0_1_85	5315,0	4,22	0,02	**5100,0**	**5100,0**	**0,00**	**0,39**	5100,0	5182,8	1,62	323,20
30_3_0_1_85	5430,0	0,74	0,02	5390,0	5431,2	0,77	2,84	5390,0	5431,4	0,77	41,20
30_2_0_2_85	5317,6	2,07	0,02	5210,0	5278,8	1,32	2,91	5210,0	5278,9	1,32	109,00
30_3_0_2_85	5430,0	0,74	0,02	5390,0	5430,1	0,74	2,72	5390,0	5430,1	0,74	10,80
30_5_0_0_95	5470,0	2,63	0,02	5330,0	5350,0	0,38	9,72	5330,0	5395,0	1,22	912,80
30_6_0_0_95	5470,0	1,11	0,02	5410,0	5467,5	1,06	9,00	5410,0	5470,0	1,11	397,80
30_3_0_1_95	5430,0	3,04	0,03	5270,0	5280,0	0,19	42,95	5270,0	5378,0	2,05	218,90
30_4_0_1_95	5470,0	1,48	0,02	5390,0	5465,2	1,40	5,08	5390,0	5467,4	1,44	99,00
30_2_0_2_95	5313,4	17,55	0,02	**4520,0**	**4520,0**	**0,00**	**0,41**	4520,0	4660,2	3,10	213,80
30_3_0_2_95	5430,0	0,74	0,02	5390,0	5425,0	0,65	2,47	5390,0	5424,9	0,65	68,10
30_4_1_48_90	5470,0	184,90	0,03	**1920,0**	**1920,0**	**0,00**	**0,02**	**1920,0**	**1921,0**	**0,05**	**13,20**
30_5_1_48_90	5470,0	127,92	0,03	**2400,0**	**2400,0**	**0,00**	**0,00**	**2400,0**	**2400,8**	**0,04**	**158,50**
30_3_1_49_90	5421,5	150,99	0,03	**2160,0**	**2160,0**	**0,00**	**0,03**	**2160,0**	**2160,6**	**0,03**	**7,80**
30_4_1_49_90	5470,0	89,93	0,02	**2880,0**	**2880,0**	**0,00**	**0,03**	**2880,0**	**2880,8**	**0,03**	**59,90**
30_5_1_50_90	5470,0	16,38	0,00	**4700,0**	**4700,0**	**0,00**	**0,05**	**4700,0**	**4701,0**	**0,02**	**1121,40**
30_6_1_50_90	5470,0	1,48	0,02	5390,0	5612,3	4,12	17,44	5390,0	5417,0	0,50	678,20
30_5_1_40_85	5470,0	79,34	0,02	**3050,0**	**3050,0**	**0,00**	**0,01**	**3050,0**	**3051,0**	**0,03**	**583,00**
30_6_1_40_85	5470,0	51,52	0,02	3610,0	3653,7	1,21	4,69	3610,0	3647,6	1,04	1962,12
30_7_1_40_85	5470,0	34,73	0,02	4060,0	4196,0	3,35	4,30	4060,0	4118,1	1,43	897,80
30_6_1_41_85	5470,0	2,63	0,00	5330,0	5340,0	0,19	6,73	5330,0	5388,5	1,10	939,00
30_7_1_41_85	5470,0	1,11	0,00	5410,0	5466,0	1,04	11,09	5410,0	5470,0	1,11	375,70
30_8_1_41_85	5470,0	0,00	0,00	**5470,0**	**5470,9**	**0,02**	**1,11**	**5470,0**	**5470,0**	**0,00**	**0,00**
30_4_1_42_85	5470,0	18,91	0,02	**4600,0**	**4600,0**	**0,00**	**0,09**	**4600,0**	**4601,0**	**0,02**	**1271,70**
30_5_1_42_85	5470,0	1,48	0,02	5390,0	5670,3	5,20	21,03	5390,0	5420,1	0,56	414,80
Média	5436,5	32,31	0,02	4533,1	4580,2	0,92	7,12	4533,1	4571,7	0,77	430,43

Tabela 4.7 - Resultados do Algoritmo de Geração de Colunas para a rede de 30 nós.

Instância	Número de *Clusters*	Algoritmo de Geração de Colunas					Solução Inteira do PMR Final		Gap GC (%)
		Número inicial de colunas	PMR inicial	Número final de colunas	PMR final	Tempo GC (s)	Solução (IP)	Tempo (IP)	
30_2_0_0_85	2	5000	3700,0	5002	3700,0	0,500	**3700,0**	0,031	0,000
30_3_0_0_85	2	5000	5390,0	5089	5396,4	27,922	5390,0	0,109	0,119
30_2_0_1_85	2	5000	5090,0	5031	5100,0	5,484	5090,0	0,063	0,196
30_3_0_1_85	2	5000	5390,0	5090	5430,0	3,656	5390,0	0,110	0,742
30_2_0_2_85	2	5000	5210,0	5085	5278,8	4,703	5210,0	0,063	1,321
30_3_0_2_85	2	5000	5422,0	5054	5430,0	1,797	5390,0	0,078	0,742
30_5_0_0_95	2	5000	5341,3	5013	5350,0	6,657	5330,0	0,250	0,375
30_6_0_0_95	2	5000	5463,9	5036	5467,0	40,328	5410,0	2,078	1,054
30_3_0_1_95	2	5000	5240,0	5027	5280,0	14,672	5240,0	0,078	0,763
30_4_0_1_95	2	5000	5450,0	5062	5465,0	29,783	5390,0	1,094	1,391
30_2_0_2_95	2	5000	4520,0	5002	4520,0	0,657	**4520,0**	0,047	0,000
30_3_0_2_95	2	5000	5390,0	5100	5423,5	11,297	5390,0	0,156	0,622
30_4_1_48_90	2	5000	1920,0	5000	1920,0	0,391	**1920,0**	0,047	0,000
30_5_1_48_90	2	5000	2392,5	5008	2400,0	2,984	**2400,0**	0,094	0,000
30_3_1_49_90	2	5000	2160,0	5000	2160,0	0,329	**2160,0**	0,032	0,000
30_4_1_49_90	2	5000	2880,0	5000	2880,0	0,406	**2880,0**	0,047	0,000
30_5_1_50_90	2	5000	4700,0	5000	4700,0	0,718	**4700,0**	0,109	0,000
30_6_1_50_90	2	5000	5406,2	5038	5411,3	47,688	5390,0	0,750	0,395
30_5_1_40_85	2	5000	3050,0	5002	3050,0	0,782	**3050,0**	0,109	0,000
30_6_1_40_85	2	5000	3626,0	5032	3646,9	30,156	3610,0	0,485	1,022
30_7_1_40_85	2	5000	4078,1	5023	4081,0	18,547	4060,0	0,375	0,517
30_6_1_41_85	2	5000	5326,2	5017	5340,0	10,719	5300,0	1,016	0,755
30_7_1_41_85	2	5000	5461,4	5038	5465,5	38,890	5350,0	2,953	2,159
30_8_1_41_85	2	5000	5470,0	5001	5470,0	1,000	**5470,0**	0,203	0,000
30_4_1_42_85	2	5000	4600,0	5001	4600,0	0,703	**4600,0**	0,110	0,000
30_5_1_42_85	2	5000	5402,1	5035	5415,0	33,577	5390,0	0,594	0,464
Média	2	5000	4541,5	5030,2	4553,1	12,859	4528,1	0,426	0,486

Tabela 4.8 - Resultados do CPLEX, heurística e metaheurísticas para a rede de 324 nós.

Instância	CPLEX			Heurística			AGC				GRASP			CS			
	Solução	Gap C	Tempo	Solução	Gap H	Tempo	Melhor	Média	Gap G	Tempo	Melhor	Média	Tempo	Melhor	Média	Gap CS	Tempo
324_10_0_0_95	**21460**	0	67,4	21386	0,34	0,14	21431	21373,0	0,14	8,63	21446	21441,20	2,05	21455	21447,2	0,023	3,16
324_10_0_1_95	**35360**	0	242,1	35250	0,31	0,16	35342	35304,0	0,05	7,79	35339	35336,50	2,06	35359	35354,6	0,003	3,17
324_10_0_2_95	**45390**	0	305,3	45300	0,20	0,20	45347	45245,0	0,09	7,81	45341	45334,60	2,01	45374	45354,6	0,035	3,10
324_10_0_0_85	**37180**	0	481	37081	0,27	0,22	37145	37069,0	0,09	8,10	37157	37155,80	2,24	37173	37163,2	0,019	3,46
324_10_0_1_85	**51000**	0	297,3	50750	0,49	0,20	50880	50711,0	0,24	7,69	50948	50946,30	2,30	50948	50946,3	0,102	3,41
324_10_0_2_85	**59740**	0	961,1	59598	0,24	0,20	59624	59437,0	0,19	7,62	59693	59688,50	2,24	59693	59688,5	0,079	3,33
324_10_1_40_85	**27700**	0	292,1	27583	0,42	0,17	27675	27602,0	0,09	8,54	27670	27666,60	2,18	27698	27692,1	0,007	3,37
324_10_1_41_85	**29360**	0	216,5	29288	0,25	0,14	29324	29260,0	0,12	8,27	29335	29330,90	2,23	29351	29341,9	0,031	3,52
324_10_1_42_85	**30950**	0	1074,1	30902	0,16	0,17	30932	30895,0	0,06	8,34	30928	30920,80	2,21	30948	30943,3	0,006	3,33
324_10_1_48_90	**26920**	0	421,6	26855	0,24	0,14	26883	26835,0	0,14	8,35	26899	26896,60	2,18	26917	26910,6	0,011	3,50
324_10_1_49_90	**28330**	0	359,1	28206	0,44	0,25	28280	28221,0	0,18	8,28	28295	28285,00	2,24	28318	28310,3	0,042	3,43
324_10_1_50_90	**29680**	0	529,9	29638	0,14	0,22	29641	29593,0	0,13	8,26	29658	29656,10	2,24	29672	29665,5	0,027	3,36
324_20_0_0_95	**42911**	0.021*	9672,3	42714	0,48	0,73	42577	42318,0	0,80	24,30	42813	42792,20	4,24	42840	42804,9	0,186	9,37
324_20_0_1_95	**70720**	0	9911,9	70368	0,50	0,74	70471	70308,0	0,35	23,40	70561	70529,80	4,43	70656	70628,2	0,090	9,06
324_20_0_2_95	**90778**	0.003	10800	90424	0,39	0,81	89970	89355,0	0,89	24,15	90550	90518,90	4,64	90556	90521,2	0,245	9,40
324_20_0_0_85	**74315**	0.061*	4791,9	73981	0,45	0,83	73407	73001,0	1,22	23,69	74165	74106,00	4,80	74165	74106,7	0,202	9,71
324_20_0_1_85	**101928**	0.071	10800	100628	1,28	1,02	99576	98353,0	2,31	24,69	101374	101177,40	4,88	101374	101177,4	0,544	9,07
324_20_0_2_85	**119445**	0.030	10800	118451	0,83	0,77	116639	115235,0	2,35	23,33	118771	118613,60	4,86	118771	118613,6	0,564	8,99
324_20_1_40_85	**55397**	0.005	10800	55006	0,71	0,84	54804	54414,0	1,07	23,92	55193	55147,40	4,63	55306	55226,7	0,164	9,65
324_20_1_41_85	**58720**	0	828,9	58577	0,24	1,02	58009	57571,0	1,21	24,70	58602	58582,60	4,72	58604	58583,4	0,198	10,33
324_20_1_42_85	**61900**	0.002	10800	61637	0,42	0,88	61545	61266,0	0,57	23,81	61746	61715,10	4,68	61847	61822,6	0,086	9,74
324_20_1_48_90	**53839**	0.002	10800	53377	0,86	0,63	53300	52958,0	1,00	24,32	53647	53607,50	4,54	53793	53689,5	0,085	9,82
324_20_1_49_90	**56651**	0.016	10800	56180	0,83	1,34	56216	55813,0	0,77	24,15	56321	56254,30	4,59	56532	56429,9	0,210	9,55
324_20_1_50_90	**59357**	0.005	10800	59119	0,40	0,94	58941	58577,0	0,70	26,03	59253	59237,00	4,74	59285	59242,6	0,121	10,08
Média	52876,3	0,009	4868,9	52595,8	0,46	0,53	52414,9	52113,1	0,62	16,17	52737,8	52705,90	3,41	52776,5	52736,0	0,128	6,46

Tabela 4.9 - Resultados das relaxações para a rede de 324 nós.

Instância	Relaxação de Programação Linear			Relaxação Lagrangeana				LagClus			
	Limite superior	Gap PL (%)	Tempo (s)	Limite inferior	Limite superior	Gap Lag (%)	Tempo (s)	Limite inferior	Limite superior	Gap LC (%)	Tempo (s)
324_10_0_0_95	124446,0	479,90	0,250	**21460,0**	**21460,0**	**0,000**	**9,9**	**21460,0**	**21460,4**	**0,002**	**27,7**
324_10_0_1_95	128311,6	262,87	0,218	**35360,0**	**35360,0**	**0,000**	**12,5**	**35360,0**	**35360,5**	**0,001**	**456,6**
324_10_0_2_95	129688,0	185,72	0,156	**45390,0**	**45390,0**	**0,000**	**3,6**	**45390,0**	**45390,8**	**0,002**	**30,7**
324_10_0_0_85	128624,9	245,95	0,188	**37180,0**	**37180,0**	**0,000**	**10,6**	**37180,0**	**37180,7**	**0,002**	**23,2**
324_10_0_1_85	130136,3	155,17	0,141	**51000,0**	**51000,0**	**0,000**	**2,2**	**51000,0**	**51000,1**	**0,000**	**4755,8**
324_10_0_2_85	130695,0	118,77	0,125	**59740,0**	**59740,0**	**0,000**	**1,8**	**59740,0**	**59740,6**	**0,001**	**2311,7**
324_10_1_40_85	126583,5	356,98	0,250	**27700,0**	**27700,0**	**0,000**	**14,9**	**27700,0**	**27700,6**	**0,002**	**26,9**
324_10_1_41_85	127049,3	332,73	0,171	**29360,0**	**29360,0**	**0,000**	**15,2**	**29360,0**	**29360,9**	**0,003**	**25,1**
324_10_1_42_85	127428,1	311,72	0,203	**30950,0**	**30950,0**	**0,000**	**15,0**	**30950,0**	**30950,5**	**0,002**	**28,3**
324_10_1_48_90	126357,7	369,38	0,219	**26920,0**	**26920,0**	**0,000**	**14,5**	**26920,0**	**26920,8**	**0,003**	**26,8**
324_10_1_49_90	126761,9	347,45	0,204	**28330,0**	**28330,0**	**0,000**	**15,0**	**28330,0**	**28330,6**	**0,002**	**26,8**
324_10_1_50_90	127126,2	328,32	0,219	**29680,0**	**29680,0**	**0,000**	**15,2**	**29680,0**	**29680,5**	**0,002**	**26,7**
324_20_0_0_95	195213,5	354,83	1,594	**42920,0**	**42920,0**	**0,000**	**10,0**	42920,0	42938,3	0,043	10800
324_20_0_1_95	197056,3	178,64	1,313	**70720,0**	**70720,0**	**0,000**	**12,5**	**70720,0**	**70720,5**	**0,001**	**8737,0**
324_20_0_2_95	197611,6	117,69	1,078	90778,0	90780,0	0,002	1036,1	90778,0	90793,9	0,018	10800
324_20_0_0_85	197198,7	165,36	1,063	74315,0	74360,0	0,061	1642,4	74315,0	74422,6	0,145	10800
324_20_0_1_85	197727,2	93,99	1,047	101928,0	102000,0	0,071	537,1	101928,0	102202,9	0,270	10800
324_20_0_2_85	197799,8	65,60	0,735	119445,0	119478,3	0,028	742,3	119445,0	120339,8	0,749	10800
324_20_1_40_85	196328,2	254,40	1,390	55397,0	55400,0	0,005	4197,2	55397,0	55441,6	0,081	10800
324_20_1_41_85	196504,9	234,65	1,343	**58720,0**	**58720,0**	**0,000**	**15,2**	58720,0	58749,4	0,050	10800
324_20_1_42_85	196661,9	217,71	1,078	**61900,0**	**61900,0**	**0,000**	**15,1**	**61900,0**	**61900,8**	**0,001**	**5844,8**
324_20_1_48_90	196241,5	264,50	1,313	53839,0	53840,0	0,002	4111,2	53839,0	53905,3	0,123	10800
324_20_1_49_90	196396,2	246,68	1,187	56651,0	56660,0	0,016	3694,2	56651,0	56770,4	0,211	10800
324_20_1_50_90	196537,3	231,11	1,390	59357,0	59360,0	0,005	4307,9	59357,0	59369,9	0,022	10800
Média	162270,2	246,67	0,703	52876,7	52883,7	0,008	852,2	52876,7	52943,0	0,072	5431,1

Tabela 4.10 - Resultados do Algoritmo de Geração de Colunas para a rede de 324 nós.

Instância	Número de *Clusters*	Algoritmo de Geração de Colunas					Solução Inteira do PMR Final		Gap GC (%)
		Número inicial de colunas	PMR inicial	Número final de colunas	PMR final	Tempo GC (s)	Solução (IP)	Tempo (IP)	
324_10_0_0_95	10	10000	21456,0	10011	21460,0	65,1	**21460,0**	0,141	0,000
324_10_0_1_95	10	10000	35357,0	10004	35360,0	47,1	**35360,0**	0,188	0,000
324_10_0_2_95	10	10000	45377,0	10023	45390,0	327,6	**45390,0**	0,203	0,000
324_10_0_0_85	10	10000	37169,0	10019	37180,0	274,9	**37180,0**	0,187	0,000
324_10_0_1_85	10	10000	50959,2	10041	51000,0	507,5	**51000,0**	0,265	0,000
324_10_0_2_85	10	10000	59682,0	10042	59740,0	604,3	**59740,0**	0,203	0,000
324_10_1_40_85	10	10000	27693,0	10019	27700,0	238,2	**27700,0**	0,172	0,000
324_10_1_41_85	10	10000	29353,5	10016	29360,0	145,7	**29360,0**	0,141	0,000
324_10_1_42_85	10	10000	30949,0	10005	30950,0	42,8	**30950,0**	0,156	0,000
324_10_1_48_90	10	10000	26914,0	10018	26920,0	144,6	**26920,0**	0,156	0,000
324_10_1_49_90	10	10000	28321,0	10022	28330,0	315,5	**28330,0**	0,172	0,000
324_10_1_50_90	10	10000	29672,0	10010	29680,0	81,7	**29680,0**	0,157	0,000
324_20_0_0_95	10	10000	42856,0	10085	42920,0	2063,8	**42920,0**	0,360	0,000
324_20_0_1_95	10	10000	70691,0	10050	70720,0	1066,8	**70720,0**	0,578	0,000
324_20_0_2_95	10	10000	90712,0	10105	90780,0	2390,7	**90780,0**	0,750	0,000
324_20_0_0_85	10	10000	74224,0	10111	74360,0	2629,5	74358,0	0,891	0,003
324_20_0_1_85	10	10000	101386,0	10263	101997,1	10800,0*	101973,0	1,328	0,024
324_20_0_2_85	10	10000	118426,0	10296	119471,2	10800,0*	119448,0	1,594	0,019
324_20_1_40_85	10	10000	55334,5	10184	55400,0	8456,6	55396,0	1,016	0,007
324_20_1_41_85	10	10000	58614,0	10076	58720,0	1377	**58720,0**	0,469	0,000
324_20_1_42_85	10	10000	61862,0	10049	61900,0	734,7	**61900,0**	0,453	0,000
324_20_1_48_90	10	10000	53784,0	10128	53840,0	4644,8	53838,0	0,688	0,004
324_20_1_49_90	10	10000	56533,0	10216	56659,2	10800,0*	56654,0	0,922	0,009
324_20_1_50_90	10	10000	59295,0	10067	59360,0	1272,9	**59360,0**	0,532	0,000
Média	10	10000	52774,2	10077,8	52883,2	2507,1	52880,6	0,501	0,003

Tabela 4.11 - Resultados do CPLEX, heurística e metaheurísticas para a rede de 818 nós.

Instância	CPLEX			Heurística			AGC				GRASP			CS			
	Solução	Gap C	Tempo	Solução	Gap H	Tempo	Melhor	Média	Gap G	Tempo	Melhor	Média	Tempo	Melhor	Média	Gap CS	Tempo
818_10_0_0_95	**21460**	0	2957,7	21429	0,14	4,94	21455	21449,3	0,02	43,65	21459	21458,3	6,30	**21460**	21459,9	0,000	11,23
818_10_0_1_95	**35360**	0	3404,1	35339	0,06	13,05	35356	35346,0	0,01	50,45	**35360**	**35360,0**	6,65	**35360**	**35360,0**	0,000	11,54
818_10_0_2_95	**45390**	0	3320,1	45375	0,03	12,09	45387	45377,4	0,01	49,37	45389	45388,2	6,97	**45390**	**45390,0**	0,000	12,17
818_10_1_48_90	**26920**	0	3004,2	26907	0,05	11,38	26915	26901,0	0,02	48,35	26918	26918,0	6,10	**26920**	**26920,0**	0,000	11,32
818_10_1_49_90	**28330**	0	3303,5	28309	0,07	8,63	28320	28298,0	0,04	47,78	28329	28328,6	6,67	**28330**	**28330,0**	0,000	12,23
818_10_1_50_90	**29680**	0	2780,8	29661	0,06	18,28	29678	29673,8	0,01	47,81	**29680**	29679,4	6,42	**29680**	**29680,0**	0,000	11,33
818_20_0_0_95	**42920**	0	4579,9	42793	0,30	25,67	42870	42817,7	0,12	76,20	42903	42900,0	14,51	**42920**	42918,9	0,000	54,84
818_20_0_1_95	**70720**	0	6392,9	70644	0,11	37,03	70653	70557,1	0,09	80,42	70717	70715,1	16,90	**70720**	70719,2	0,000	62,91
818_20_0_2_95	**90780**	0	7098,1	90730	0,06	75,19	90747	90672,9	0,04	84,99	90772	90769,5	19,18	**90780**	90779,6	0,000	66,80
818_20_0_0_85	**74360**	0	6993,7	74313	0,06	75,05	74341	74279,4	0,03	90,04	74353	74352,1	16,27	**74360**	74359,8	0,000	66,81
818_20_0_1_85	100989	435,000	10800	101933	-0,93	76,06	101955	101898,8	-0,96	89,19	101996	101991,9	19,24	**102000**	101998,9	-1,001	67,36
818_20_0_2_85	119405	352,480	10800	119397	0,01	105,48	119445	119306,4	-0,03	89,18	119468	119466,0	21,26	**119480**	119477,6	-0,063	74,36
818_20_1_40_85	55398	0,004	10800	55325	0,13	66,11	55341	55235,7	0,10	88,33	55396	55395,3	15,93	**55400**	55399,0	-0,004	61,04
818_20_1_41_85	58719	0,002	10800	58637	0,14	69,63	58706	58677,6	0,02	87,62	58711	58709,2	15,24	**58720**	58719,9	-0,002	57,31
818_20_1_42_85	61818	774,000	10800	61814	0,01	71,81	61839	61719,7	-0,03	90,43	61894	61892,2	16,16	**61900**	61898,8	-0,133	58,71
818_20_1_48_90	**53840**	0	5565,0	53728	0,21	22,41	53814	53762,2	0,05	87,87	53827	53825,7	15,35	**53840**	53839,6	0,000	59,56
818_20_1_49_90	**56660**	0	5254,6	56602	0,10	34,34	56605	56465,3	0,10	88,71	56653	56651,6	16,11	**56660**	**56660,0**	0,000	67,42
818_20_1_50_90	**59360**	0	4919,3	59269	0,15	39,84	59336	59279,8	0,04	87,04	59352	59350,2	15,02	**59360**	59359,9	0,000	58,48
818_50_0_0_85	185876	0,013	10800	185426	0,24	756,72	184428	184153,6	0,78	177,34	185779	185755,8	50,62	**185880**	185775,6	-0,002	364,20
818_50_0_1_85	251029	115,230	10800	254509	-1,39	730,20	253438	252884,6	-0,96	171,86	254860	254836,6	58,37	**254985**	254905,0	-1,576	387,37
818_50_0_2_85	292912	84,450	10800	298217	-1,81	757,73	296763	296182,8	-1,31	171,74	298547	298511,2	63,58	**298582**	298517,6	-1,936	392,29
818_50_1_48_90	**134598**	0,001	10800	134088	0,38	596,11	134079	133999,4	0,39	177,67	134474	134444,5	43,99	**134598**	134561,6	0,000	335,00
818_50_1_49_90	**141648**	0,001	10800	141281	0,26	571,17	140439	140143,4	0,85	174,96	141548	141527,5	46,32	141586	141532,8	0,044	356,80
818_50_1_50_90	**148398**	0,001	10800	147931	0,31	667,67	147202	147123,8	0,81	173,96	148280	148253,1	46,62	148383	148321,9	0,010	337,40
Média	91107,1	73,380	7432,3	91402,4	-0,05	201,94	91213	91091,9	0,01	98,96	91527,7	91520,0	22,91	91553,9	91536,9	-0,190	124,90

Tabela 4.12 - Resultados das relaxações para a rede de 818 nós

Instância	Relaxação de Programação Linear			Relaxação Lagrangeana				LagClus			
	Limite superior	Gap PL (%)	Tempo (s)	Limite inferior	Limite superior	Gap Lag (%)	Tempo (s)	Limite inferior	Limite superior	Gap LC (%)	Tempo (s)
818_10_0_0_95	510850,9	2280,5	750,2	*	*	*	*	**21460,0**	21460,9	0,004	2230,5
818_10_0_1_95	513291,1	1351,6	1361,4	*	*	*	*	**35360,0**	35360,8	0,002	3763,1
818_10_0_2_95	514036,2	1032,5	903,9	*	*	*	*	**45390,0**	45390,8	0,002	4143,7
818_10_1_48_90	512159,6	1802,5	1440,8	*	*	*	*	**26920,0**	26920,9	0,003	3328,9
818_10_1_49_90	512389,8	1708,7	1558,4	*	*	*	*	**28330,0**	28330,8	0,003	3359,00
818_10_1_50_90	512591,8	1627,1	1305,8	*	*	*	*	**29680,0**	29680,9	0,003	3136,4
818_20_0_0_95	540286,0	1158,8	338,2	*	*	*	*	**42920,0**	42920,5	0,001	2435,2
818_20_0_1_95	540286,0	663,9	333,0	*	*	*	*	70720,0	98879,3	39,818	10800
818_20_0_2_95	540286,0	495,2	373,7	*	*	*	*	90780,0	105775,5	16,519	10800
818_20_0_0_85	540286,0	626,6	400,2	*	*	*	*	**74360,0**	74360,5	0,001	2505,9
818_20_0_1_85	540286,0	435,0	258,3	*	*	*	*	102000,0	109518,6	7,371	10800
818_20_0_2_85	540286,0	352,5	291,4	*	*	*	*	119479,0	120383,3	0,757	10800
818_20_1_40_85	540286,0	875,3	352,6	*	*	*	*	55400,0	55432,9	0,059	10800
818_20_1_41_85	540286,0	820,1	429,8	*	*	*	*	**58720,0**	58720,7	0,001	3142,8
818_20_1_42_85	540286,0	774,0	408,2	*	*	*	*	61900,0	61980,5	0,130	10800
818_20_1_48_90	540286,0	903,5	362,7	*	*	*	*	**53840,0**	53840,7	0,001	2232,8
818_20_1_49_90	540286,0	853,6	341,3	*	*	*	*	56660,0	56729,8	0,123	10800
818_20_1_50_90	540286,0	810,2	446,7	*	*	*	*	59360,0	59621,1	0,440	10800
818_50_0_0_85	540286,0	190,6	162,3	*	*	*	*	185876,0	445932,5	139,909	10800
818_50_0_1_85	540286,0	115,2	101,7	*	*	*	*	254851,0	540286,0	112,001	10800
818_50_0_2_85	540286,0	84,5	35,7	*	*	*	*	298516,0	506904,0	69,808	10800
818_50_1_48_90	540286,0	301,4	429,5	*	*	*	*	134598,0	240260,5	78,502	10800
818_50_1_49_90	540286,0	281,4	294,3	*	*	*	*	141648,0	273443,9	93,045	10800
818_50_1_50_90	540286,0	264,1	434,3	*	*	*	*	148398,0	319535,2	115,323	10800
Média	533352,8	825,4	546,4					91548,6	142152,9	28,076	7561,6

Tabela 4.13 - Resultados do algoritmo de geração de colunas para a rede de 818 nós.

Instância	Número de *Clusters*	Algoritmo de Geração de Colunas					Solução Inteira do PMR Final		Gap GC (%)
		Número inicial de colunas	PMR inicial	Número final de colunas	PMR final	Tempo GC (s)	Solução (IP)	Tempo (IP)	
818_10_0_0_95	20	20000	21460,0	20007	21460,0	162,8	**21460,0**	0,266	0,000
818_10_0_1_95	20	20000	35360,0	20011	35360,0	207,2	**35360,0**	0,516	0,000
818_10_0_2_95	20	20000	45390,0	20006	45390,0	250,5	**45390,0**	0,796	0,000
818_10_1_48_90	20	20000	26920,0	20009	26920,0	184,9	**26920,0**	0,469	0,000
818_10_1_49_90	20	20000	28330,0	20008	28330,0	195,6	**28330,0**	0,453	0,000
818_10_1_50_90	20	20000	29680,0	20008	29680,0	198,9	**29680,0**	0,391	0,000
818_20_0_0_95	20	20000	42920,0	20008	42920,0	247,3	**42920,0**	0,781	0,000
818_20_0_1_95	20	20000	70720,0	20021	70720,0	388,9	**70720,0**	1,094	0,000
818_20_0_2_95	20	20000	90780,0	20008	90780,0	459,0	**90780,0**	1,234	0,000
818_20_0_0_85	20	20000	74360,0	20006	74360,0	411,1	**74360,0**	1,610	0,000
818_20_0_1_85	20	20000	102000,0	20011	102000,0	445,5	**102000,0**	1,563	0,000
818_20_0_2_85	20	20000	119480,0	20005	119480,0	472,1	**119480,0**	1,390	0,000
818_20_1_40_85	20	20000	55400,0	20006	55400,0	292,0	**55400,0**	1,671	0,000
818_20_1_41_85	20	20000	58720,0	20012	58720,0	371,3	**58720,0**	0,953	0,000
818_20_1_42_85	20	20000	61900,0	20016	61900,0	379,5	**61900,0**	1,110	0,000
818_20_1_48_90	20	20000	53840,0	20017	53840,0	341,6	**53840,0**	1,453	0,000
818_20_1_49_90	20	20000	56660,0	20008	56660,0	323,1	**56660,0**	1,546	0,000
818_20_1_50_90	20	20000	59360,0	20014	59360,0	363,2	**59360,0**	1,609	0,000
818_50_0_0_85	20	20000	185838,1	20058	185900,0	3677,3	185898,0	18,640	0,001
818_50_0_1_85	20	20000	254950,3	20143	255000,0	5055,2	**255000,0**	17,578	0,000
818_50_0_2_85	20	20000	298581,6	20099	298700,0	5938,5	298490,0	66,063	0,070
818_50_1_48_90	20	20000	134581,8	20048	134600,0	2312,4	**134600,0**	5,203	0,000
818_50_1_49_90	20	20000	141551,9	20054	141650,0	2705,5	**141650,0**	8,312	0,000
818_50_1_50_90	20	20000	148338,3	20034	148400,0	2440,8	**148400,0**	3,813	0,000
Média	20	20000	91546,8	20025,7	91563,8	1159,4	91554,9	5,771	0,003

4.8 Considerações Finais

Este capítulo apresentou os resultados da aplicação de heurística, metaheurística, relaxações e método de decomposição para o PPLAMC. Três formulações foram apresentadas para esse problema, feitas para permitir a conveniente aplicação de diferentes métodos de solução. Uma delas, a $\overline{PPLAMC}$ permitiu a modelagem deste problema em grafo de conflitos. Entretanto, a utilização desse modelo implica em relaxar muito mais restrições que o problema modelado por grafo de cobertura. Desta forma, este último foi usado para permitir aplicação da LagClus e do algoritmo de geração de colunas.

Várias relaxações foram consideradas para o PPLAMC: a relaxação de programação linear, a relaxação lagrangeana tradicional e a LagClus. A primeira apresentou os piores resultados, chegando a mais de 800% de gap médio para as instâncias de 818 pontos. A relaxação lagrangeana tradicional apresentou bons resultados para instâncias de 30 e 324 pontos. Entretanto, não conseguiu executar a primeira iteração do algoritmo de subgradientes após três horas de processamento para instâncias de 818 pontos. Conseqüentemente, nenhum gap pôde ser calculado, deixando a LagClus como a relaxação mais interessante.

A abordagem do CS parece ser uma boa alternativa de solução para o PPLAMC, considerando os resultados médios e os tempos de processamento. Entretanto, a abordagem de geração de colunas pareceu ser a melhor estratégia para o PPLAMC, pois apresentou os melhores resultados médios entre todos os relatados, em tempos computacionais aceitáveis, além de provar os valores ótimos de várias instâncias em que o CPLEX não conseguiu em três horas de processamento.

5 CONCLUSÃO

Este trabalho discutiu e propôs métodos de solução para dois problemas de localização de facilidades, o UFLP e o PPLAMC, baseados em técnicas alternativas, dado que esses problemas são *NP-Hard*.

Foram discutidos os conceitos de grafos e particionamento de grafos, que juntamente com as relaxações e decomposição formam a base para a Relaxação Lagrangeana com *Clusters* (LagClus) e do Algoritmo de Geração de Colunas, que são duas das técnicas alternativas empregadas na solução desses problemas. Além dessas, foram empregadas também outras relaxações e, especificamente para a solução do PPLAMC, as metaheurísticas Algoritmo Genético Construtivo (AGC) e *Clustering Search* (CS). As principais contribuições e as próximas etapas deste trabalho são apresentadas a seguir.

5.1 Resumo das Contribuições

A relaxação lagrangeana tradicional consiste em identificar e relaxar um subconjunto de restricões do problema, de tal modo que o problema relaxado resultante seja de mais fácil solução. Para a otimização do dual lagrangeano pode ser empregado o algoritmo de subgradientes. Dependendo do problema, ao relaxar no sentido lagrangeano um subconjunto completo de restrições, a relaxação resultante pode não gerar bons limitantes. Uma alternativa para esse inconveniente é modelar as restrições do problema por meio de um grafo, particioná-lo em *clusters* (agrupamento de vértices bem definidos) e relaxar as arestas entre *clusters* no sentido lagrangeano. Esta é a idéia da LagClus, que, por não relaxar o conjunto completo de restrições apresenta vantagem sobre relaxação lagrangeana tradicional (RIBEIRO, 2007). Esta idéia foi aplicada ao UFLP em instâncias de difícil solução para métodos baseados em relaxação linear e para instâncias de fácil solução obtidas da *OR-Library*. Para isso, foi utilizado um grafo de conflitos, montado a partir do complemento das variáveis de localização, que forneceu melhores resultados que a relaxação de programação linear e a lagrangeana tradicional.

Por decompor o problema original em *clusters*, o Algoritmo de Geração de Colunas também pode ser aplicado ao UFLP, em que o PMR é formado pelas restrições de conexão (acomplamento) e os subproblemas geradores de colunas, formados nos *clusters*. O Algoritmo de Geração de Colunas apresentou melhor estratégia, se comparada à LagClus e as demais relaxações, por apresentar limites duais de melhor qualidade, apesar de acréscimos nos tempos computacionais.

Outra contribuição foi a resolução do Problema Probabilístico de Localização-alocação de Máxima Cobertura (PPLAMC) usando-se também a LagClus e o Algoritmo de Geração de

Colunas. Três formulações foram apresentadas para esse problema, feitas para permitir a conveniente aplicação de diferentes métodos de solução. Uma delas, a $\overline{PPLAMC}$, permitiu a modelagem do problema em grafo de conflitos. Entretanto, a utilização desse modelo implica relaxar muito mais restrições que o problema modelado por grafo de cobertura. Dessa forma, esse último foi usado para permitir aplicação da LagClus e do Algoritmo de Geração de Colunas. A relaxação de programação linear e a relaxação lagrangeana tradicional também foram consideradas para o PPLAMC, sendo que a lagrangeana tradicional empregou a formulação $PPLAMC^{MG}$. Como a primeira apresentou os piores resultados, fornecendo *gap* médio maior que 800% para as instâncias de 818 pontos, e a segunda não conseguiu executar a primeira iteração do algoritmo de subgradientes após três horas de processamento para as instâncias de 818 pontos, a LagClus se apresentou como a relaxação mais interessante.

Da mesma forma que na aplicação ao UFLP, a decomposição do problema por meio de *clusters* permitiu a utilização da decomposição de Dantzig-Wolfe e, conseqüentemente, pôde-se aplicar um algoritmo de geração de colunas ao PPLAMC. Essa solução foi mais adequada que a LagClus, pois para todas as instâncias que o CPLEX encontrou solução ótima o algoritmo de geração de colunas também encontrou, com melhores tempos computacionais. A geração de colunas também provou a otimalidade de outras seis instâncias não provadas pelo CPLEX em três horas de processamento. A média de *gaps* para a abordagem de geração de colunas, cerca de 0,003%, foi insignificante se comparada às relaxações e ao CPLEX, que chegou a mais de 100% em várias instâncias, mesmo após três horas de processamento.

A terceira contribuição deste trabalho é a resolução do PPLAMC com as metaheurísticas Algoritmo Genético Construtivo (AGC) e o *Clustering Search* (CS). Os resultados médios refletem 50 execuções de cada instância e apresentam melhores valores que o CPLEX para alguns casos. Entre as duas, a abordagem do CS é a melhor solução e mostra-se como uma excelente alternativa quando o fator tempo de processamento é o mais crítico requisito. Entretanto, entre todas as opções, a abordagem de Geração de Colunas pareceu ser a melhor estratégia para o PPLAMC, pois apresentou os melhores resultados médios entre todos os relatados, em tempos computacionais aceitáveis, além de provar os valores ótimos de várias instâncias no tempo permitido, instâncias que falhou o CPLEX.

Assim, as principais contribuições deste trabalho são:

- Aplicação da LagClus e do Algoritmo de Geração de Colunas à solução do Problema de Localização de Facilidades Não-capacitado (*Uncapacitated Facility Location Problem - UFLP*), utilizando-se a abordagem do grafo de conflitos;

- Resolução do Problema Probabilístico de Localização-alocação de Máxima Cobertura (PPLAMC) com as metaheurísticas Algoritmo Genético Construtivo (AGC) e o *Clustering Search* (CS); e
- Aplicação da LagClus e do Algoritmo de Geração de Colunas à solução do Problema Probabilístico de Localização-alocação de Máxima Cobertura, utilizando-se a abordagem do grafo de cobertura.

5.2 Trabalhos Futuros

A pesquisa iniciada neste trabalho pode ser complementada, considerando-se alguns aspectos que demandam investigações mais detalhadas. Com isso, novos trabalhos e novas linhas de pesquisa podem ser criados:

- Para o Algoritmo de Geração de Colunas, o estudo de novas formas de gerar o conjunto inicial de colunas e de investigar a sua influência na obtenção de soluções duais de melhor qualidade e na melhoria do tempo de processamento. Além disso, pode-se considerar a busca de obtenção de um tratamento mais eficiente para a eliminação das colunas improdutivas e do uso de técnicas de estabilização (SENNE et al., 2007) para a redução do tempo computacional. Pode-se ainda estudar a aplicação do algoritmo *Branch and Price* para os problemas em que a solução ótima não foi encontrada (SENNE et al., 2005);
- Investigação da aplicação da LagClus e do Algoritmo de Geração de Colunas a outros problemas de localização de facilidades com restrições modeladas por grafos;
- Avaliação da possibilidade da modelagem de problemas de localização usando-se outros tipos de grafos, visando à redução da quantidade de restrições relaxadas;
- Para esses novos grafos e mesmo para os empregados (conflitos e cobertura), a pesquisa pode ser direcionada para a possibilidade da aplicação da clonagem de vértices sugerida por Sachdeva (2004), que consiste em copiar, em cada um dos *clusters*, os vértices das arestas de ligação entre eles, de tal forma que as restrições correspondentes a essas arestas não sejam relaxadas, mas divididas nos subproblemas sem relaxação. Serão relaxadas apenas as novas restrições que garantam a igualdade entre as variáveis originais e suas cópias.

REFERÊNCIAS BIBLIOGRÁFICAS

ALP, O.; ERKUT, E.; DREZNER, Z. An efficient genetic algorithm for the p-median problem. **Annals of Operations Research**, v. 122, n. 1–4, p. 21–42, 2003. 38

ALVES, M. L.; ALMEIDA, M. T. Simulated annealing algorithm for the simple plant location problem: A computational study. **Revista Investigação Operacional**, v. 12, 1992. 34

ATAMTÜRK, A.; NEMHAUSER, G. L.; SAVELSBERGH, M. W. P. Conflict graphs in solving integer programming problems. **European Journal of Operational Research**, v. 121, n. 1, p. 40–55, 2000. 24, 28, 29

BAFNA, V.; BANSAL, V. The number of recombination events in a sample history: conflict graph and lower bounds. **IEEE Transations on Computational Biology and Bioinformatics**, v. 1, n. 2, p. 78–90, April-June 2004. 29

BALAKRISHNAN, V. K. **Schaum's outline of theory and problems of graph theory**. New York: McGraw-Hill, 1997. 293 p. ISBN 0-07-005489-4. 27

BALAS, E.; CARRERA, M. A dynamic subgradient-based branch-and-bound procedure for set covering. **Operations Research**, v. 44, p. 875–890, 1996. 36

BAUTISTA, J. P. J. A grasp algorithm to solve the unicost set covering problem. **Computers & Operations Research**, v. 34, n. 10, p. 3162–3173, October 2007. 36

BAZARAA, M. S.; JARVIS, J. J.; SHERALI, H. D. **Linear programming and network flows**. New York: John Wiley & Sons, 1990. 2nd edition. 43

BEASLEY, J. E. Or-library: distributing test problems by eletronic mail. **Journal of Operational Research Society**, v. 41, p. 1069–1072, 1990. Disponível em: <http://people.brunel.ac.uk/~mastjjb/jeb/info.html>. 36, 57, 67

_____. Lagrangean heuristics for location problems. **European Journal of Operational Research**, v. 65, p. 383–399, 1993. 24, 33

BEASLEY, J. E.; JøRNSTEN, K. Enhancing an algorithm for set covering problems. **European Journal of Operational Research**, v. 58, p. 293–300, 1992. 36

BJORNDAL, M. H.; CAPRARA, A.; COWLING, P. I.; CROCE, F. D.; LOURENÇO, H.; MALUCELLI, F.; ORMAN, A.; PISINGER, D.; REGO, C.; SALAZAR, J. J. Some thoughts on combinatorial optimization. **European Journal of Operational Research**, v. 83, p. 253–270, 1995. 23

BOZKAYA, B.; ERKUT, E.; LAPORTEZ, G. A tabu search heuristic and adaptive memory procedure for political districting. **European Journal of Operational Research**, v. 144, n. 1, p. 12–26, 2003. 38

BROTCORNE, L.; LAPORTE, G.; SEMET, F. Fast heuristics for large scale covering-location problems. **Computers & Operations Research**, v. 29, n. 6, p. 651–665, May 2002. 36

BUI, T. N.; MOON, B. R. Genetic algorithm and graph partitioning. **IEEE Transactions on Computers**, v. 45, n. 7, p. 841–855, 1996. 31

BURKE, E. K.; ELLIMAN, D. G.; WEARE, R. F. A university timetabling system based on graph colouring and constraint manipulation. **Journal of Research on Computing in Education**, v. 27, p. 1–18, 1994. 29

CAPRARA, M. F. A.; TOTH, P. Algorithms for the set covering problem. **Annals of Operations Research**, v. 98, p. 353–371, 2000. 36

CERIA, S.; NOBILI, P.; SASSANO, A. Set covering problem. In: DELL'AMICO, M.; MAFFIOLI, F.; MARTELLO, S. (Ed.). **Annotated bibliographies in combinatorial optimization**. Chichester: John Wiley & Sons, 1997. p. 415–428. 36

_____. A lagrangian-based heuristic for large-scale set covering problems. **Mathematical Programming**, v. 81, p. 215–228, 1998. 36

CHARTRAND, G.; OELLERMANN, O. R. **Applied and algorithmic graph theory**. New York: McGraw-Hill, 1992. 432 p. ISBN 0-075-57101-3. 27

CHAVES, A. A.; CORRÊA, F. A.; LORENA, L. A. N. Clustering search heuristic for the capacitated p-median problem. In: ABRAHAM, A.; CORCHADO, E.; CORCHADO, J. M. (Ed.). **Innovations in hybrid intelligent systems**. Salamanca: Springer, 2008. (Advances in Soft Computing, v. 44), p. 136–143. ISBN 978-3-540-74971-4. 39

CHIYOSHI, F. Y.; GALVÃO, R. D. A statistical analysis of simulated annealing applied to the p-median problem. **Annals of Operations Research**, v. 96, p. 61–74, 2000. 38

CHRISTOFIDES, N. **Graph theory**: an algorithmic approach. London: Academic Press, 1975. 415 p. ISBN 0-121-74350-0. 27

CHUNG, C. H. Recent applications of the maximal covering location problem (mclp) model. **Journal of the Operational Research Society**, v. 37, p. 735–746, 1986. 37

CHURCH, R. L.; REVELLE, C. Maximal covering location problem. **Papers of the Regional Science Association**, v. 32, p. 101–118, 1974. 37, 75, 76

CORNUÉJOLS, G.; FISHER, M. L.; NEMHAUSER, G. L. Location of bank accounts to optimize float: an analytic study of exact and approximate algorithms. **Management Science**, v. 23, n. 8, p. 789–810, 1977. 33, 38

CORNUÉJOLS, G.; NEMHAUSER, G. L.; WOLSEY, L. A. The uncapacitated facility location problem. In: MIRCHANDANI, P. B.; FRANCIS, R. L. (Ed.). **Discrete Location Theory**. New York: Wiley-Interscience, 1990. cap. 3, p. 119–171. 27, 33, 34

CORNUÉJOLS, G.; THIZY, J. M. Some facets of the simple plant location polytope. **Mathematical Programming**, v. 23, p. 50–74, 1982. 57

CORRÊA, F. A.; CHAVES, A. A.; LORENA, L. A. N. Heurística híbrida com detecção de regiões promissoras aplicada ao problema probabilístico de localização-alocação de máxima cobertura. In: SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL, 39., 2007, Fortaleza. **Anais...** Rio de Janeiro: Sociedade Brasileira de Pesquisa Operacional, 2007. p. 1553–1565. 87, 104

_____. Hybrid heuristics for the probabilistic maximal covering location-allocation problem. **Operational Research - An International Journal**, Special Issue on: New Research Advances on Facility Location, 2008. Accepted. 81, 104

CORRÊA, F. A.; LORENA, L. A. N. Aplicação da relaxação lagrangeana e do algoritmo genético construtivo na solução do problema probabilístico de localização-alocação de máxima cobertura. **Gestão & Produção**, v. 13, n. 2, p. 233–244, 2006. 43, 81, 87, 104

CORRÊA, F. A.; LORENA, L. A. N.; RIBEIRO, G. M. A decomposition approach for the probabilistic maximal covering location-allocation problem. **Computers and Operations Research**, 2008. Submitted. 24

CORRÊA, F. A.; LORENA, L. A. N.; SENNE, E. L. F. Lagrangean relaxation with clusters for the uncapacitated facility location problem. In: CONGRESO LATINO-IBEROAMERICANO DE INVESTIGACIÓN OPERATIVA, 13., 2006, Montevideo. **Proceedings...** Montevideo: Universidad de la República, 2006a. ISBN 9974-7699-9-X. 29, 33

_____. Método de geração de colunas aplicado ao problema de localização de facilidades não-capacitado. In: SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL, 38., 2006, Goiânia. **Anais...** Rio de Janeiro: Sociedade Brasileira de Pesquisa Operacional, 2006b. p. 1648–1657. 29, 33

_____. A column generation approach for the uncapacitated facility location problem. **European Journal of Operational Research**, 2008. Submitted. 29

CURRENT, J. R.; O'KELLY, M. Locating emergency warning sirens. **Decision Sciences**, v. 23, n. 1, p. 221–234, 1992. 37

DANTZIG, G. B.; WOLFE, P. Decomposition principle for linear programs. **Operations Research**, v. 8, p. 101–111, 1960. 45

DASKIN, M. S. **Network and discrete location: models, algorithms and applications**. New York: John Wiley & Sons, 1995. 25, 27, 33, 35, 37

EATON, D.; HECTOR, M.; SANCHEZ, V.; LATINGUA, R.; MORGAN, J. Determining ambulance deployment in santo domingo, dominican republic. **Journal of the Operational Research Society**, v. 37, p. 113–126, 1986. 37

ERLENKOTTER, D. A dual-based procedure for uncapacitated facility location. **Operations Research**, v. 26, p. 992–1009, 1978. 33

ESPEJO, G. A.; GALVÃO, R. D. O uso das relaxações lagrangeana e surrogate em problemas de programação inteira. **Pesquisa Operacional**, v. 22, n. 3, p. 387–402, 2002. 40, 42

FEO, T.; RESENDE, M. G. C. Greedy randomized adaptive search procedures. **Journal of Global Optimization**, v. 6, p. 109–133, 1995. 96

FISHER, M.; KEDIA, P. Optimal solution of set covering/partitioning problems using dual heuristics. **Management Science**, v. 36, p. 674–688, 1998. 36

FJÄLLSTRÖM, P. O. **Algorithms for graph partitioning**: a survey. Computer and Information Science, 1998. V. 3,n. 10. Disponível em: <http://www.ep.liu.se/ea/cis/1998/010>. 31

FLESZAR, K.; HINDI, K. S. An effective vns for the capacitated p-median problem. **European Journal of Operational Research**, v. 191, n. 3, p. 612–622, December 2008. 38

FOGLIATTI, M. C.; MATTOS, N. M. C. **Teoria de filas**. Rio de Janeiro: Interciência, 2007. 75

FURTADO, J. C. **Algoritmo genético construtivo na otimização de problemas combinatoriais de agrupamentos**. 112 p. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos-SP, Brasil, 1998. 49, 52, 94, 95

GALVÃO, R. D. Uncapacitated facility location problems: contributions. **Pesquisa Operacional**, v. 24, n. 1, p. 7–38, April 2004. 33, 37, 39

GALVÃO, R. D.; REVELLE, C. S. A lagrangean heuristic for the maximal covering location problem. **European Journal of Operational Research**, v. 88, n. 1, p. 114–123, 1996. 37

GALVÃO, R. D.; SANTIBANEZ-GONZALEZ, E. del R. A lagrangean heuristic for the p_k-median dynamic location problem. **European Journal of Operational Research**, v. 58, p. 250–262, 1992. 38

GAREY, M.; JOHNSON, D.; STOCKMEYER, L. Some simplified np-complete graph problems. **Theoretical Computer Science**, v. 1, p. 237–267, 1976. 31

GEN, M.; TSUJIMURA, Y.; ISHIZAKI, S. Optimal design of a star-lan using neural networks. **Computers and Industrial Engineering**, v. 31, p. 855–859, 1996. 34

GHOSH, D. Neighborhood search heuristics for the uncapacitated facility location problems. **European Journal of Operational Research**, v. 150, n. 1, p. 150–162, October 2003. 34

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. 31

GLOVER, F. Tabu search and adaptive memory programming: Advances, applications and challenges. **Interfaces in Computer Science and Operations Research**, p. 1–75, 1996. 97

GOLDBERG, D. E. **Genetic algorithms in search, optimization and machine learning**. New York: Addison-Wesley, 1989. 49

HAKIMI, S. L. Optimum location of switching centers and the absolute centers and medians of graph. **Operations Research**, v. 12, p. 450–459, 1964. 28, 38

_____. Optimum distribution of switching centers and some graph related theoretic problem. **Operations Research**, v. 13, p. 462–475, 1965. 38

HALE, T. S.; MOBERG, C. R. Location science review. **Annals of Operations Research**, v. 123, p. 21–35, 2003. 33, 37

HELD, M.; KARP, R. M. The traveling salesman problem and minimum spanning trees. **Operations Research**, v. 18, p. 1138–1162, 1970. 41, 103

HOFFMAN, K.; PADBERG, M. W. Solving airline crew-scheduling problems by branch-and-cut. **European Journal of Operational Research**, v. 39, p. 667–682, 1993. 29

HOUGLAND, E. S.; STEPHENS, N. T. Air pollutant monitor sitting by analytical techniques. **Journal of the Air Pollution Control Association**, v. 26, p. 52–53, 1976. 37

HUISMAN, D. A column generation approach for the rail crew re-scheduling problem. **European Journal of Operational Research**, v. 180, n. 1, p. 163–173, July 2007. 36

ILOG. **Ilog Cplex 7.5: Reference manual**. France: [s.n.], 2001. 610 p. 68

IRANI, S.; LEUNG, V. Scheduling with conflicts on bipartite and intervals graphs. **Journal of Scheduling**, v. 6, p. 287–307, 2003. 29

JACOBS, L. W.; BRUSCO, M. J. A local-search heuristic for large set-covering problems. **Naval Research Logistics: an International Journal**, v. 42, n. 7, p. 1129–1140, 1995. 36

JANS, R.; DEGRAEVE, Z. A note on a symmetrical set covering problem: The lottery problem. **European Journal of Operational Research**, v. 186, n. 1, p. 104–110, April 2008. 36

JÄRVINEN, P.; RAJALA, J.; SINERVO, H. A branch-and-bound algorithm for seeking the p-median. **Operations Research**, v. 20, p. 173–178, 1972. 38

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. 31

KARISCH, S. E.; RENDL, F.; CLAUSEN, J. Solving graph bisection problems with semidefinite programming. **INFORMS Journal on Computing**, v. 12, n. 3, p. 177–191, 2000. 31

KARYPIS, G.; KUMAR, V. A fast and high quality multilevel scheme for partitioning irregualar graphs. **SIAM Journal on Scientific Computing**, v. 20, p. 359–392, 1998. 31, 32, 100

_____. Multilevel k-way partitioning scheme for irregular graphs. **Journal of Parallel and Distributed Computing**, v. 48, n. 1, p. 96–129, 1998. 31, 32

KERNIGHAN, B. W.; LIN, S. An efficient heuristic procedure for partitioning graphs. **The Bell System Technical J.**, v. 49, p. 291–307, 1970. 31

KOCHETOV, Y.; IVANENKO, D. Computationally dificult instances for the uncapacitated facility location problem. In: METAHEURISTICS INTERNATIONAL CONFERENCE, 5., 2003, Kioto. **Proceedings...** Kioto, 2003. p. 41–45. 14, 57, 67, 68, 71

KÖRKEL, M. On the exact solution of large-scale simple plant location problem. **European Journal of Operational Research**, v. 39, p. 157–173, 1989. 33

KOSKOSIDIS, Y. A.; POWELL, W. B. Clustering algorithms for consolidation of customer orders into vehicle shipments. **Transportation Research**, v. 26B, n. 5, p. 365–379, October 1992. 38

KRATICA, J.; TOSIC, D.; FILIPOVIC, V.; LJUBIC, I. Solving the simple plant location problem by genetic algorithm. **RAIRO Operations Research**, v. 39, p. 127–142, 2001. 34

KUEHN, A. A.; HAMBURGUER, M. J. A heuristic program for location warehouses. **Management Science**, v. 9, p. 643–649, 1963. 33

LACERDA, E. G. M.; CARVALHO, A. C. P. L. F. de. Introdução aos algoritmos genéticos. In: CONGRESSO NACIONAL DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 19., 1999, Rio de Janeiro. **Anais...** Rio de Janeiro, 1999. p. 51–126. 49

LAN, G.; DEPUY, G. W.; WHITEHOUSE, G. E. An effective and simple heuristic for the set covering problem. **European Journal of Operational Research**, v. 29, n. 3, p. 1387–1403, February 2007. 36

LARSON, R. C.; ODONI, A. R. **Urban operations research**. Englewood Cliffs: Prentice Hall, 1981. 76

LORENA, L. A. N.; FURTADO, J. C. Constructive genetic algorithm for clustering problems. **Evolutionary Computation**, v. 9, n. 3, p. 309–327, 2001. 38

LORENA, L. A. N.; LOPES, F. B. A surrogate heuristic for set covering problems. **European Journal of Operational Research**, v. 79, p. 138–150, 1994. 36

LORENA, L. A. N.; LOPES, L. S. Genetic algorithms applied to computationally difficult set covering problems. **Journal of the Operational Research Society**, v. 48, p. 440–445, 1997. 36

LORENA, L. A. N.; PEREIRA, M. A. A lagrangean/surrogate heuristic for the maximal covering location problem using hillsman´s edition. **International Journal of Industrial Engineering**, v. 9, p. 57–67, 2002. 37, 84, 96, 98

LORENA, L. A. N.; SENNE, E. L. F. Local search heuristics for capacitated p-median problems. **Networks and Spatial Economics**, v. 3, p. 407–419, 2003. 38, 41, 82, 83

_____. A column generation approach to capacitated p-median problems. **Computers and Operations Research**, v. 31, n. 6, p. 863–876, 2004. 38

MARIANOV, V. Location of multiple-server congestible facilities for maximizing expected demand, when services are non-essential. **Annals of Operations Research**, v. 123, p. 125–141, 2003. 35

MARIANOV, V.; REVELLE, C.; SNYDER, S. Selecting compact habitat reserves for species with differential habitat size needs. **Computers & Operations Research**, v. 35, p. 475–487, 2008. 36

MARIANOV, V.; SERRA, D. Probabilistic, maximal covering location-allocation models for congested systems. **Journal of Regional Science**, v. 38, n. 3, p. 401–424, 1998. 37, 75, 76, 77, 78, 80, 81, 82, 87, 88, 96, 99, 101, 102

_____. Hierarchical location-allocation models for congested systems. **European Journal of Operational Research**, v. 135, n. 1, p. 195–208, November 2001. 37, 75, 76

MARTELLO, S.; TOTH, P. **Knapsack problems**: algorithms and computer implementations. [S.l.]: John Wiley & Sons Inc, 1990. 308 p. ISBN 0-471-92420-2. 83

MEDARD, C. P.; SAWHNEY, N. Airline crew scheduling from planning to operations. **European Journal of Operational Research**, v. 183, n. 3, p. 1013–1027, December 2007. 36

MICHEL, L.; HENTENRYCK, P. V. A simple tabu search for warehouse location. **European Journal of Operational Research**, v. 157, n. 3, p. 576–591, September 2003. 34

MLADENOVIC, N.; BRIMBERG, J.; HANSEN, P.; MORENO-PÉREZ, J. A. The p-median problem: a survey of metaheuristic approaches. **European Journal of Operational Research**, v. 179, n. 3, p. 927–939, June 2007. 39

MONFROGLIO, A. Hybrid heuristic algorithms for set covering. **Computers and Operations Research**, v. 25, n. 6, p. 441–455, June 1998. 36

MOORE, G. C.; REVELLE, C. S. The hierarchical service location problem. **Management Science**, v. 28, n. 7, p. 775–780, 1982. 37

MURRAY, A. L.; GERRARD, R. A. Capacitated service and regional constraints in location-allocation modeling. **Location Science**, v. 5, n. 2, p. 103–118, 1997. 81

NARCISO, M. G. **A relaxação lagrangeana/surrogate e algumas aplicações em otimização combinatória**. 134 p. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos-SP, Brasil, 1998. 43

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. 24, 83, 103

NETTO, P. O. B. **Grafos: teoria, modelos, algoritmos**. São Paulo: Edgard Blücher, 1996. 27

OLIVEIRA, A. C. M. **Algoritmos evolutivos híbridos com detecção de regiões promissoras em espaços de busca contínuo e discreto**. 200 p. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2004. 50, 53

OLIVEIRA, A. C. M.; LORENA, L. A. N. Detecting-promising areas by evolutionary clustering search. **Advances in Artificial Intelligence Series**, v. 3171, p. 385–394, 2004. 50, 53

PAPADIMITRIOU, C. H.; STEIGLITZ, K. **Combinatorial optimization - algorithms and complexity**. New York: Dover Publications, Inc., 1998. 496 p. ISBN 0-486-40258-4. 23

PARKER, R. G.; RARDIN, R. L. **Discrete optimization**. New York: Academic Press, 1988. 472 p. ISBN 0-12-545075-3. 41, 42, 83, 103

PEREIRA, M. A. **Um método *branch-and-price* para problemas de localização de p-medianas**. 90 p. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos-SP, Brasil, 2005. 38

PEREIRA, M. A.; LORENA, L. A. N.; SENNE, E. L. F. A column generation approach for the maximal covering location problem. **International Transactions in Operations Research**, v. 14, p. 349–364, 2007. 37

PIRKUL, H. An integer programming model for the allocation of databases in a distributed computer system. **European Journal of Operational Research**, v. 26, n. 3, p. 401–411, 1986. 33

_____. Efficient algorithms for the capacitated concentrator location problem. **Computers and Operations Research**, v. 14, n. 3, p. 197–208, 1987. 38

PIRKUL, H.; SCHILLING, D. A. The maximal covering location problem with capacities on total workload. **Management Science**, v. 37, n. 2, p. 233–248, 1991. 41, 82

REEVES, C. R. **Modern heuristic techniques for combinatorial problems**. London: McGraw-Hill Book Company, 1995. 42

RESENDE, M.; WERNECK, R. **A GRASP with path-relinking for the *p*-median problem**. AT&T Labs Research, 2002. Technical Report TD-5E53XL. 38

RESENDE, M. G. C.; WERNECK, R. F. A hybrid multistart heuristic for the uncapacitated facility location problem. **European Journal of Operational Research**, v. 174, n. 1, p. 54–68, October 2006. 34, 67

REVELLE, C. S.; EISELT, H. A. Location analysis: A synthesis and survey. **European Journal of Operational Research**, v. 165, n. 1, p. 1–19, August 2005. 33

RIBEIRO, G. M. **Relaxação lagrangeana com divisão em *clusters* para alguns problemas de otimização modelados em grafos de conflitos**. 194 p. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos-SP, Brasil, 2007. 24, 32, 43, 44, 45, 49, 88, 93, 119

RIBEIRO, G. M.; LORENA, L. A. N. Lagrangean relaxation with clusters and column generation for the manufacturer's pallet loading problem. **Computers and Operations Research**, v. 34, n. 9, p. 2695–2708, 2007. 29, 43, 106

_____. Lagrangean relaxation with clusters for point-feature cartographic label placement problems. **Computers and Operations Research**, v. 35, n. 7, p. 2129–2140, 2008. 29, 43, 106

_____. Optimizing the woodpulp stowage using lagrangean relaxation with clusters. **Journal of the Operational Research Society**, v. 59, p. 600–606, 2008. 29, 43

ROLLAND, E.; PIRKUL, H.; GLOVER, F. Tabu search for graph partitioning. **Annals of Operations Research**, v. 63, p. 209–232, 1996. 31

SACHDEVA, S. **Development of a branch and price approach involving vertex cloning to solve the maximum weighted independent set problem**. Master's thesis — Texas A&M University, Texas, EUA, 2004. 121

SENNE, E. L. F.; LORENA, L. A. N. Lagrangean/surrogate heuristics for p-median problems. In: LAGUNA, M.; GONZALEZ-VELARDE, J. (Ed.). **Computing tools for modeling, optimization and simulation: interfaces in computer science and operations research**. Boston: Kluwer Academic Publishers, 2000. p. 115–130. 38

SENNE, E. L. F.; LORENA, L. A. N.; PEREIRA, M. A. A branch-and-price approach to p-median location problems. **Computers & Operations Research**, v. 32, n. 6, p. 1655–1664, 2005. 38, 121

_____. A simple stabilizing method for column generation heuristics: an application to p-median location problems. **International Journal of Operations Research**, v. 4, n. 3, p. 1–9, 2007. 121

SERRA, D.; MARIANOV, V. **New trends in public facility location modelling**. Barcelona, Spain, 2004. Economics and Business Working Paper 755. Disponível em: <http://www.econ.upf.edu/docs/papers/downloads/755.pdf>. 33, 37

SIMON, H. D. Partitioning of unstructured problems for parallel processing. **Computing Systems in Engineering**, v. 2, n. 2–3, p. 135–148, 1991. 31

SINGH, K. N. The uncapacitated facility location problem: some applications in scheduling and routing. **International Journal of Operations Research**, v. 5, n. 1, p. 36–43, July 2008. 33

SUN, M. Solving the uncapacitated facility location problem using tabu search. **Computers & Operations Research**, v. 33, n. 9, p. 2563–2589, September 2006. 34, 67

TEITZ, M. B.; BART, P. Heuristics methods for estimating the generalized vertex median of a weighted graph. **Operations Research**, v. 16, p. 955–961, 1968. 38

VAITHYANATHAN, S.; BURKE, L.; MAGENT, M. Massively parallel analog tabu search using neural networks applied to simple plant location problems. **European Journal of Operational Research**, v. 93, p. 317–330, 1996. 34

WARRIER, D.; WILHELM, W. E.; WARREN, J. S.; HICKS, I. V. A branch-and-price approach for the maximum weight independent set problem. **Networks**, v. 46, n. 4, p. 198–209, December 2005. 43

WEST, D. B. **Introduction to graph theory**. Upper Saddle River: Prentice Hall, 2000. 470 p. ISBN 0-130-14400-2. 27

WILSON, R. J.; WATKINS, J. J. **Graphs**: an introductory approach. New York: John Wiley & Sons, 1990. 352 p. ISBN 0-471-61554-4. 27

WOLSEY, L. A. **Integer programming**. New York: John Wiley & Sons, 1998. 264 p. ISBN 0-471-28366-5. 40, 45, 103

PUBLICAÇÕES TÉCNICO-CIENTÍFICAS EDITADAS PELO INPE

Teses e Dissertações (TDI)

Teses e Dissertações apresentadas nos Cursos de Pós-Graduação do INPE.

Manuais Técnicos (MAN)

São publicações de caráter técnico que incluem normas, procedimentos, instruções e orientações.

Notas Técnico-Científicas (NTC)

Incluem resultados preliminares de pesquisa, descrição de equipamentos, descrição e ou documentação de programas de computador, descrição de sistemas e experimentos, apresentação de testes, dados, atlas, e documentação de projetos de engenharia.

Relatórios de Pesquisa (RPQ)

Reportam resultados ou progressos de pesquisas tanto de natureza técnica quanto científica, cujo nível seja compatível com o de uma publicação em periódico nacional ou internacional.

Propostas e Relatórios de Projetos (PRP)

São propostas de projetos técnico-científicos e relatórios de acompanhamento de projetos, atividades e convênios.

Publicações Didáticas (PUD)

Incluem apostilas, notas de aula e manuais didáticos.

Publicações Seriadas

São os seriados técnico-científicos: boletins, periódicos, anuários e anais de eventos (simpósios e congressos). Constam destas publicações o Internacional Standard Serial Number (ISSN), que é um código único e definitivo para identificação de títulos de seriados.

Programas de Computador (PDC)

São a seqüência de instruções ou códigos, expressos em uma linguagem de programação compilada ou interpretada, a ser executada por um computador para alcançar um determinado objetivo. Aceitam-se tanto programas fonte quanto os executáveis.

Pré-publicações (PRE)

Todos os artigos publicados em periódicos, anais e como capítulos de livros.