versão impressa ISSN 0101-7438 / versão online ISSN 1678-5142

O PROBLEMA DO CARTEIRO CHINÊS, ALGORITMOS EXATOS E UM AMBIENTE MVI PARA ANÁLISE DE SUAS INSTÂNCIAS: SISTEMA XNÊS

Marcos José Negreiros Gomes*
Wiler Rodrigues Coelho Júnior
Augusto Wagner de Castro Palhano
Emanuel Ferreira Coutinho
Gerson Alves de Castro
Universidade Estadual do Ceará (UECE)
Fortaleza – CE, Brasil
negreiro@graphvs.com.br

Francisco José Negreiros Gomes
Gianfrancesca Cutini Barcellos
Bruno Fernandes Rezende
Lúcio Wagner Lessa Pereira
Universidade Federal do Espírito Santo (UFES)
Vitória – ES, Brasil
bfrezende@hotmail.com

* *Corresponding author* / autor para quem as correspondências devem ser encaminhadas

Recebido em 07/2007; aceito em 03/2009 após 1 revisão
Received July 2007; accepted March 2009 after one revision

Resumo

Apresenta-se um estudo geral sobre o Problema do Carteiro Chinês (PCC), nas versões simétrica, orientada e mista, do ponto de vista dos algoritmos exatos até então publicados sobre o assunto. Para apresentar as soluções exatas das versões do problema, foram utilizadas as implementações exatas dos algoritmos de Sherafat (dos casos orientado e misto) com adaptações, e de Edmonds & Johnson (do simétrico) adaptada de Burkard & Derigs. O trabalho também apresenta as diferenças de comportamento dos métodos em situações bastante peculiares, através de um conjunto de instâncias testes com o objetivo de apurar mais detalhadamente o seu desempenho. Por fim, mostramos o *software* (Sistema **XNÊS**) idealizado para gerar grafos instâncias (simétricos, orientados e/ou mistos) e soluções exatas para estas versões do Problema do Carteiro Chinês (PCC). Implementa-se aqui uma versão de um modelo de geração de grafos no modo de Modelagem Visual Interativa (MVI), através de um editor específico de grafos com atributos e *informações de efeitos visuais* nos vértices e nas ligações. Este *software* pode ser usado como demonstração efetiva da importância do problema na academia, assim como um potencial analista de rotas de problemas semelhantes, ou até mesmo como verificador de instâncias das mais complicadas ou de razoável porte do PCC.

Palavras-chave: problema do carteiro chinês; grafos; modelagem visual interativa.

Abstract

We do a general study over the Chinese Postman Problem (CCP), in the symmetric, oriented and mixed cases, in consideration to the modeling and exact algorithms published most recently. To find exact solutions "for/to" these versions of the problem we show and use exact implementations reported by Sherafat (to the Oriented and the Mixed cases) with adaptations, Edmonds and Johnson (symmetric case) adapted from Burkard & Derigs approach. The work also presents a set of evaluation instances to the methods, and peculiarities behind them to the performance and solution quality analysis. In this work we show the software **XNÊS**, which was created to generate graph instances (pure symmetric, mixed and pure oriented) for the Chinese Postman Problem (CPP). We show an implementation of a graph generator in a Visual Interactive Modeling (MVI) mode, which builds graphs with attributes and bindings on nodes and links. This software can be used to demonstrate the effective importance of this problem and its application to academia, as its potential to solve other arc routing problems, or to verify difficult and/or high scale CPP instances.

Keywords: Chinese postman problem; graphs; visual interactive modeling.

1. Introdução

Os problemas de Percurso em Arcos são dos mais antigos relacionados a grafos. A primeira referência que se conhece sobre eles vem do famoso problema das sete pontes de Königsberg, figura 1. Buscava-se saber se havia um caminho fechado que atravessasse exatamente uma vez sete pontes sobre o rio Pregel em Köningsberg, hoje Kaliningrad/ Rússia. O problema foi solucionado pelo matemático suíço Leonhard Euler, que encontrou as condições para a existência de um percurso fechado (grafo euleriano), e mostrou que não havia solução que satisfizesse aquele caso particular (Euler, 1736), figura 2. Nasce então a Teoria dos Grafos.

Figura 1 – Visualização de Köningsberg, hoje Kaliningrad-Rússia desde 1946, e as sete pontes sobre o rio Pregel.

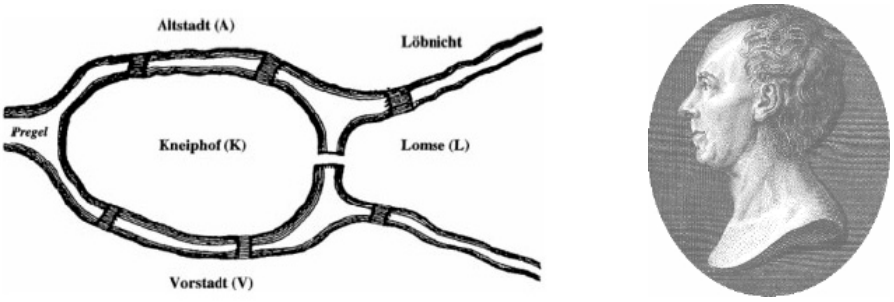

Figura 2 – Grafismo proposto por Euler, 1736 (e seu busto), representando a situação do problema das sete pontes. Os quatro vértices representam Altstadt e Löbnicht (A) localizadas sobre a margem norte do Rio Pregel, Vorstadt (V) localizada na margem sul, e as duas ilhas Kneiphof (K) e Lomse (L).

Em 15 de abril de 2007 comemorou-se 300 anos do nascimento de Euler (1707-1883). Sua importância histórica como matemático vai desde a criação da Teoria dos Grafos, a se tornar uma das maiores referências de sua época. Já em seu tempo Pierre-Simon Laplace (1749-1827) dizia, "*leiam Euler, leiam Euler, ele é o mestre de todos nós*". Euler foi na verdade um dos mais produtivos matemáticos da história, publicou mais de 25 livros e ensaios, e 850 artigos técnicos. A comunidade científica comemorou através de um grande congresso em Junho de 2007, em São Pitersburgo na Rússia, patrocinado pelo Governo Russo, a Academia Russa de Ciências, a Fundação Euler e muitas outras organizações científicas, este importante personagem da história da humanidade. A comunidade de Pesquisa Operacional também faz esta homenagem através de recente número especial da revista Networks (Assad, 2007; Gribkovskaia *et al.*, 2007).

Como na época a preocupação de Euler fora exclusivamente sobre a existência do caminho fechado (desafio de Konigsberga), figura 3, a questão de exibir tal percurso foi resolvida 137 anos mais tarde por Heierholzer, devido às propriedades de conectividade e unicursalidade para se ter um percurso euleriano necessitarem ser ainda provadas (Heierholzer, 1873). Muitos anos mais tarde, em 1962, um matemático da Universidade Normal de Shangtun, Kwan Mei-Ko, quando de sua passagem como funcionário dos correios durante a revolução cultural chinesa, preocupou-se com uma situação semelhante à de Euler e Heierholzer, porém adequada ao percurso dos carteiros que atendiam ruas de sua cidade. Neste caso, Kwan mostrou-se interessado em definir além da travessia, a forma mais econômica de fazê-la, realizando com a menor distância possível o percurso. Kwan definiu assim o problema: *Um carteiro tem que cobrir seu local de trabalho, antes de retornar ao posto. O problema é encontrar a menor distância de percurso para o carteiro* (Kwan, 1962; Gribkovskaia *et al.*, 2007).

Figura 3 – Transformação das sete pontes em um grafo por Euler.

O termo percurso em arcos foi estabelecido mais claramente anos mais tarde, em 1995 (Eiselt *et al.*, 1995i), com dois trabalhos sobre divisões do Problema do Carteiro Chinês, e o estado da arte em algoritmos sobre o assunto. A princípio, a notação considera um grafo G como um conjunto formado por vértices e ligações, $G=(V,L)$. O conjunto das ligações pode ser ampliado para uma dupla de conjuntos, $L=(E,A)$, onde E são denominadas as ligações não orientadas (elos) entre vértices de G, e A é o conjunto das ligações orientadas (arcos) entre vértices de G.

A classificação variada dos diversos tipos de problemas de percursos em arcos, abrevia-se no interesse deste trabalho, as principais abordagens do Problema do Carteiro Chinês (Eiselt *et al.*, 1995i) são:

1. **Problema do Carteiro Chinês – Caso Simétrico (CPP)**, onde se deseja gerar um percurso de custo mínimo sobre um grafo $G = (V, E)$, valorado e conexo, a partir de um vértice $v_0 \in V$, origem;
2. **Problema do Carteiro Chinês – Caso Dirigido (DCPP)**, onde se deseja gerar um percurso de custo mínimo sobre um grafo $G = (V, A)$, valorado e fortemente conexo (*f*-conexo), a partir de um vértice $v_0 \in V$, origem;
3. **Problema do Carteiro Chinês – Caso Misto (MCPP)**, onde se deseja gerar um percurso de custo mínimo sobre um grafo $G = (V, E, A)$, valorado e fortemente conexo (*f*-conexo), a partir de um vértice $v_0 \in V$, origem;
4. **Problema do Carteiro Chinês – Caso Íngreme (WPP)**, onde se deseja gerar um percurso de custo mínimo sobre um grafo $G = (V, E)$, valorado e conexo, a partir de um vértice $v_0 \in V$, origem, onde os elos de *E* podem ter custos distintos de travessia de *i* para *j* (c_{ij}) e de *j* para *i* (c_{ji}).

Basicamente, os problemas resumem-se em transformar o grafo original em um grafo euleriano para cada uma das versões acima, porém, para que se garanta que o grafo final seja euleriano, este deve gozar da seguinte propriedade (Eiselt *et al.*, 1995i; Nobert & Picard, 1996):

Propriedade da Unicursalidade

Seja G um grafo *f-conexo.* G é dito *unicursal* ou *euleriano* se existe um caminho fechado em G contendo cada aresta apenas uma vez e cada vértice pelo menos uma vez. As condições necessárias e suficientes para que um grafo *f-conexo* seja euleriano são dadas como segue:

1. Se *G* é não orientado (simétrico), todo vértice deve ter grau par, ou seja, um número par de elos incidentes – Teorema de Euler.
2. Se *G* é orientado, o número de arcos entrando e saindo de cada vértice é igual – Teorema de Ford & Fulkerson (Ford & Fulkerson, 1962).
3. Se *G* é misto, todo vértice em *S* deve conter um número par de arcos orientados a ele ligados; além disso, para todo conjunto $S \subseteq V$, a diferença entre o número de arcos de *S* para *V*-*S* e o número de arcos de *V*-*S* para *S* deve ser menor do que ou igual ao número de elos ligando *S* e (*V*-*S*) – Condições de balanceamento (Nobert & Picard, 1996).

Apesar de a literatura já ter proposto métodos exatos para todas as quatro variações acima colocadas, as versões do problema do carteiro chinês nos casos dirigido/orientado e misto, têm correntes de pensamento bastante diferentes e interessantes na sua resolução. As versões do caso simétrico limitam-se aos algoritmos de 1-*matching* (Edmonds & Johnson, 1973; Christofides, 1973; Püetz, 1973; Derigs, 1988). As versões do caso orientado consideram a formulação de fluxo em redes de custo mínimo e/ou uma transformação do problema de transporte (Bodin & Beltrami, 1974; Yaxiong & Yongchang, 1988; Evans & Minieka, 1978). Estas duas versões são pois polinomiais. Para o caso misto, por se tratar de uma versão NP-Difícil do PCC, temos algoritmos nas versões de programação linear inteira (Nobert & Picard, 1996), de fluxo em redes de custo mínimo e B&B combinatório (Sherafat, 1994), de fluxo com ganhos (Minieka, 1979).

Já para o caso íngreme (PCCI), por se tratar de uma generalização do problema do Carteiro Chinês, portanto também NP-Difícil, tem uma formulação proposta inicialmente para grafos totalmente simétricos, e consequentemente um algoritmo Branch & Cut, e com uma formulação estendida para o problema de particionamento de conjuntos proposta por Yan & Thompson (Grotschel & Win, 1992; Yan & Thompson, 1998).

O problema do carteiro chinês íngreme, fora proposto originalmente como o percurso sobre um grafo simétrico conexo e valorado positivamente nas arestas, e qualquer de suas versões (mista ou orientada) pode ser abordada da mesma maneira, como simétrica, bastando colocar com valor infinito (M muito grande) o custo no sentido contrário do elo correspondente a um arco de uma rede mista ou orientada, a qual será transformada na rede simétrica.

Pouco se tem feito ultimamente sobre as versões do PCC, uma vez que os casos simétrico e orientado são resolvidos por algoritmos bastante conhecidos da PO, e polinomiais. Já para o caso misto, por se tratar de um problema de natureza NP-Difícil, os trabalhos recentes têm explorado a natureza do politopo formado pelas restrições desta versão, assim como a natureza deste através de um problema estendido do Carteiro, mais restrito que o PCC (Martinez, 2003).

O Problema do Carteiro Rural (PCR) considera que o percurso deve ser garantido sobre um subconjunto de ligações do grafo, partindo e retornando de um vértice origem dado (Lenstra & Rinooy Kan, 1981; Christofides *et al.*, 1984; Corberán, 1994; Corberán, 1998; Corberán *et al.*, 2001; Nobert & Picard, 1996; Eiselt *et al.*, 1995ii; Negreiros, 1996). Deixamos para o leitor mais interessado, o estudo do Problema do Carteiro Íngreme e o Problema do Carteiro Rural, através da bibliografia referenciada. Não é objetivo deste trabalho, tratar esses problemas, apesar de já termos inserido em versão de testes do XNES um método de resolução do PCR e mais recentemente estarmos desenvolvendo estudos sobre o PCR e PCRI com a mesma base de algoritmos que apresentaremos a seguir (Negreiros, Coutinho & Palhano, 2003; Negreiros & Coelho Jr, 2007).

Neste trabalho, abordamos os métodos e propriedades existentes na resolução de cada subproblema acima colocado, e em seguida apresentamos um sistema computacional, que chamamos de **Sistema XNÊS**, onde foi construído um ambiente gráfico para a criação de instâncias e resolução de problemas do Carteiro Chinês sobre elas. Vários *softwares* já foram construídos os quais tratam da resolução deste problema do ponto de vista simétrico ou orientado ou misto, uma lista deles poderá ser encontrada em Eiselt *et al.* (1995ii). Alguns *softwares*, porém, tratam especificamente de problemas de otimização em grafos, pex., o Problema do Carteiro Chinês, como TransCAD, SisGRAFO, GOBLIN e outros. O *software* GOBLIN já possui uma versão de visualização de soluções para o caso simétrico, porém ao que se percebe, somente o SisGRAFO tem implementado chamadas para tratamento do problema em suas diferentes formas, haja vista sua especialidade. Apesar disto, no SisGRAFO não estão implementados os métodos exatos do Carteiro, o que é feito de forma exclusiva no sistema XNÊS (Negreiros, 1996; TRANSCAD, 2007; GOBLIN, 2009).

Apesar do problema parecer estar ligado a uma aplicação com foco específico, o PCC também se aplica no percurso de varredores de rua, leituristas de energia, teste topológico de sistemas computacionais em diferentes níveis, determinação de estados mínimos de energia em vidros *spin*, na minimização de vias em projetos de circuitos VLSI, na coleta de lixo, na remoção de neve, no corte de peças tais como o vidro e roupas, no corte de metal para fabricação de peças mecânicas, dentre vários outros (Negreiros Gomes, 1990; Negreiros Gomes, 1996; Eiselt *et al.*, 1995ii). A figura 4 mostra um exemplo destes tipos de problemas

relacionados diretamente ao PCC, no corte de peças sobre uma peça maior de vidro. Vemos que a partir do um desenho (padrão de corte, *layout*) de conjunto de peças menores sobre ela posicionadas, deseja-se fazer as peças menores a partir do corte feito com um braço de robô. Supondo que se trata de um corte não guilhotinado, as trilhas entre peças mostram um grafo simétrico (para este caso) onde o braço deve percorrer de modo a realizar o processo de corte automático. As listas mais grossas mostram o movimento (animação) realizado. Numa produção em escala a solução do percurso do carteiro otimiza o tempo de produção das peças.

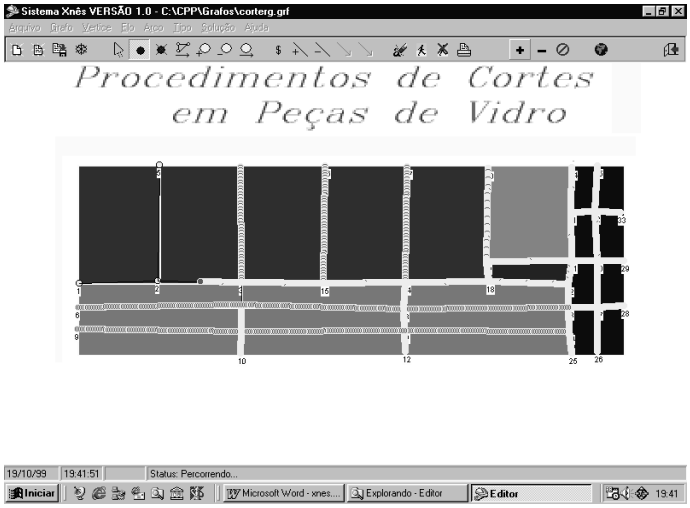

Figura 4 – Aplicação do PCC no corte automático de peças em um plano.

Este texto está organizado do seguinte modo, na seção seguinte descrevemos o problema simétrico, considerando principalmente a propriedade topológica do equilíbrio de grau entre os vértices, assim como o algoritmo geral de Edmonds & Johnson (Edmonds & Johnson, 1973) e o algoritmo para determinação de um percurso euleriano. Na seção 3 colocamos o problema do Carteiro Chinês orientado, assim como suas propriedades fundamentais de fluxo sobre os vértices, colocando também o algoritmo de circulação. Na seção 4 abordamos o problema do carteiro em grafos mistos, onde reproduzimos as propriedades de unicursalidade e o algoritmo de Sherafat (Sherafat, 1988). Na seção 5, apresentamos ensaios computacionais sobre instâncias particulares do problema e instâncias genéricas. Na seção 6, descrevemos o sistema XNÊS, colocando o gerador de grafos como uma ferramenta para a criação de instâncias dos mais variados tipos, a resolução destas através do algoritmo apropriado, gerador de multigrafos, e a visualização dos resultados a partir de editores e animação gráfica.

2. Versão Simétrica do PCC

No caso simétrico do PCC, as ligações são valoradas de modo que o percurso entre quaisquer pares de vértices de um grafo conexo é o mesmo de *i* para *j*, e de *j* para *i*. Neste caso, um percurso euleriano somente poderá ser realizado, quando um dado grafo G for transformado em um multigrafo G' de tal modo que os vértices de grau ímpar de G, possuam grau par em G'. Ou melhor, todo vértice de G, tem um número par de elos ligados a eles em G'.

Formulando-se o problema matematicamente (Edmonds & Johnson, 1973), temos:

(PCC-Simétrico) Minimizar $\sum_{ij \in E} c_{ij} x_{ij}$ [2.1]

Sujeito a:

$$\sum_{(v_i, v_j) \in E(S)} x_{ij} \geq 1\,,\ S \subset V, S \text{ são os vértices de grau ímpar} \quad [2.2]$$

$$x_{ij} \in Z_+\,,\ \forall (v_i, v_j) \in E(S) \quad [2.3]$$

onde,

Qualquer conjunto próprio S, não vazio, de V, como $E(S) = \{(v_i, v_j) \mid v_i \in S, v_j \in V - S$ ou $v_i \in V - S,\ v_j \in S\}$;

x_{ij} é o número de vezes (inteiro) que um elo de G será repetido em G';

Como resultado interessante desta formulação, Edmonds & Johnson mostraram que o poliedro formado pelas equações [2.2] e [2.3], é igual à envoltória convexa das soluções do problema, sendo que as equações [2.2]. referem-se a desigualdades de *blossons* (circuitos com 2*k*+1 vértices, os quais têm *k* elos emparelhados em um *matching* fixo) (Eiselt *et al.*, 1995i).

Assim um algoritmo para o PCC-Simétrico pode ser resumido como segue.

Procedure *Simetrico* $(G, G', p(G'))$ – (Edmonds & Johnson, 1973)
// Algoritmo genérico para o PCC-Simétrico
// Entrada: G – Grafo (V, E) valorado nas arestas e simétrico
// Saída: G´ – Grafo Euleriano Mínimo (grafo contendo as arestas repetidas de G)
p(G´) – função perímetro do grafo aumentado

Passo 1: Definir $S \subset V$, conjunto dos vértices de grau ímpar;

Passo 2: Encontrar o grafo completo $(K_{|S|})$ valorado com os percursos mínimos entre dois quaisquer vértices de S;

Passo 3: Aplicar *1-MatchingValorado* $(S, E(S), G')$ – retorna G';

Passo 4: $p(G') = \sum_{ij \in E'} c_{ij} x_{ij}$ { perímetro do multigrafo aumentado, onde E' é o conjunto dos elos de G' }.

Para formar um ciclo euleriano no grafo G', pode-se inicialmente aplicar o algoritmo acima proposto e em seguida encontrar o percurso sobre G', como segue:

Procedure *c-Euler* – (Bukard & Derigs, 1980)
// Algoritmo genérico para o percurso euleriano.
// Entrada: G´(V,L') multigrafo aumentado de G, GR[v_i] grau de v_i
// Saída: Γ, K[v_i] número de ligações visitadas em v_i,
01. Begin
02. NoAtual:=NoInicial:=NoPartida;
03. FimPercurso:=true;
04. Γ_d:={NoAtual}; // Caminho direto
05. Γ_d;:=φ; // Caminho inverso
06. **While not** FimPercurso **do**
07. begin
08. // Fase de construção de circuitos
09. Repeat
10. ProxNo:=L[NoAtual].Prox; // toma da lista de nos de NoAtual o próximo nó
11. K[NoAtual]:= K[NoAtual] +1;
12. Γ_d:= Γ_d * {ProxNo}; // guarda no conjunto caminho direto
13. **Until** NoAtual=NoInicial;
14.
15. // Fase de separação de circuitos
16. **If** | Γ_d | + |Γ_i | < |L'| **then**
17. begin
18. J:={K[v_i] | |K[v_i] < GR[v_i], $\forall v_i \in \Gamma_d$}; // seleciona um vértice ainda não percorrido totalmente
19. **If** J≠φ **then** // no caminho direto
20. begin
21. s:=posição de J[1] em Γ_d; // pega a posição do vértice no caminho direto
22. **For** i:=s+1 **to** | Γ_d | **do** Γ_i:= Γ_d[i]; // transfere do caminho direto para o inverso
23. |Γ_d|:=s; // atualiza o tamanho do conjunto do caminho direto
24. Continue;
25. end
26. J:={K[v_i] | |K[v_i] < GR[v_i], $\forall v_i \in \Gamma_i$}; // seleciona um vértice ainda não percorrido totalmente
27. **begin** // no caminho inverso
28. p := posição de J[|J|] em Γ_i; // pega a posição do vértice no caminho inverso
29. **For** i:= | Γ_i | **downto** p **do** Γ_d:= Γ_i[i]; // transfere do caminho inverso para o direto
30. |Γ_i|:=p; // atualiza o tamanho do conjunto caminho inverso
31. end
32. end
33. **else** FimPercurso:=false;
34. **end**;
35. **For** i:= | Γ_i | **downto** 1 **do** Γ_d:= Γ_i[i]; // transfere do caminho inverso para o direto
36. *Result*:= Γ_d;
37. **End**; // *c-Euler*

Os algoritmos para a resolução do PCC-Simétrico são da ordem de $O(n\,m)$, onde n é o número de vértices de grau ímpar e m é o número de elos de G, quando for utilizado um algoritmo de *1-matchingvalorado* proposto por Edmonds & Johnson (Bukard & Derigs, 1980). Já para o algoritmo do ciclo euleriano, não será necessário mais que $O(m)$, onde m é o número de elos de G'.

3. Versão Orientada do PCC

No caso orientado/dirigido, o PCC pode ser resolvido a partir da determinação do multigrafo euleriano de $G=(V,A)$, onde G é f-conexo, através de um algoritmo de fluxo de custo mínimo ou circulação de custo mínimo em uma rede, ou através de uma transformação de G num grafo para o problema de transporte (Edmonds & Johnson, 1973; Bodin & Beltrami, 1974; Yaxiong & Yongchang, 1988).

Seja I o conjunto dos vértices v_i, onde $d^+(v_i) > d^-(v_i)$ e daí ligados a um novo vértice f denominado de fonte; e seja J o conjunto dos vértices v_j onde $d^-(v_j) > d^+(v_j)$, e daí ligados a um vértice s, o qual se denomina de sumidouro. Sendo $d^+(v_i)$ definido como o número de arcos saindo e $d^-(v_i)$ o número de arcos entrando em v_i. A formulação do PCC-Orientado pode ser descrita como segue (Eiselt *et al.*, 1995i):

(PCC-Orientado) Minimizar $\sum_{v_i \in I} \sum_{v_j \in J} c_{ij} x_{ij}$ [3.1]

Sujeito a:

$$\sum_{v_j \in J} x_{ij} = d^+(v_i) - d^-(v_i)\,,\ \forall v_i \in I \qquad [3.2]$$

$$\sum_{v_i \in I} x_{ij} = d^-(v_j) - d^+(v_j)\,,\ \forall v_j \in J \qquad [3.3]$$

$$x_{ij} \geq 0\,,\ \forall v_i \in I, \forall v_j \in J \qquad [3.4]$$

A formulação acima é definida como o Problema Clássico de Transportes (Hitchcock, 1941; Koopmans, 1947; Ahuja *et al.*, 1993).

Para resolver instâncias do PCC-Orientado, nossa preocupação foi de definir um procedimento genérico que o tratasse assim como a versão mista. Esse algoritmo pode ser implementado em versões distintas para o Problema de Custo de Fluxo Mínimo, via algoritmos primal-dual (*Out-of-Kilter*), cuja complexidade pode chegar a $O(m^3)$ (Orlin, 1984), ou por outros métodos em $O(m^2 \log n)$ (Plotkin & Tardos, 1990).

Procedure *PCC-Orientado* (G, G');
// Algoritmo genérico para o PCC-Orientado

Passo 1: Tomar G, com as informações de distância, e testar sua *f*-conectividade.

Passo 2: Se *f*-conexo, transformar as informações de distância para informações de fluxo (figura 5), com os limites de fluxo máximo e mínimo permitidos por arco, e aplicar um algoritmo de fluxo em redes de custo mínimo, *Out-of-Kilter* $(G, G', Custo)$ (Bazaraa *et al.*, 1990).

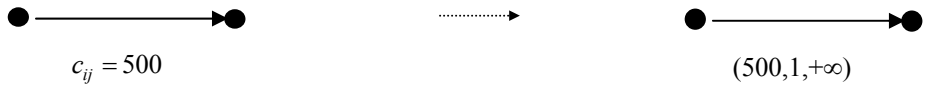

Figura 5 – Transformação de atributos de distância em atributos de fluxo e custo.

Passo 3: Após o passo e obtenção de G', onde os resultados de fluxo indicam a quantidade de vezes que cada arco será repetido em G', aplicar o resultado encontrado para formação da rota euleriana, usando o algoritmo *c-euler*.

4. Versão Mista do PCC

A versão mista do PCC contém uma generalização dos grafos não-orientados, onde existem um conjunto de elos e arcos não vazios. Em contraste com o PCC Orientado e não Orientado (simétrico), que podem ser resolvidos em tempo polinomial. Papadimitriou mostrou que a versão de decisão do PCC Misto, torna o problema $\mathcal{NP}$–completo, mesmo se restringirmos a entrada utilizando um grafo misto planar, com cada vértice com grau máximo três e todos os custos dos arcos iguais a um. Ele provou isso utilizando uma transformação do problema 3-SAT. Lenstra & Rinooy Kan também reforçam este ponto de vista um pouco depois para a classe de problemas combinatórios exclusivamente ligados a problemas de roteamento de veículos (Lenstra & Rinooy Kan, 1981; Papadimitriou, 1976).

Seja $G = (V, L)$ um grafo fortemente conexo, onde $L = (E, A)$, sendo $E = \{(v_i, v_j) : v_i, v_j \in V\} \neq \emptyset$ e $A = \{<v_s, v_k> : v_s, v_k \in V\} \neq \emptyset$, e G_m o grafo aumentado de G, e G_m euleriano. G é par se o número de elos e arcos incidentes a todo vértice for par. G é simétrico se para cada vértice o número de arcos entrando é igual ao número de arcos saindo. Um grafo é balanceado, se as condições de balanceamento e unicursalidade são satisfeitas.

Se G é par e simétrico, então ele é balanceado, porém a simetria não é uma condição suficiente a unicursalidade. Se já se sabe que G é euleriano, pode-se ter o problema de determinação de um circuito euleriano em G. Isto pode ser atingido em três fases:

1. Atribuir sentido a alguns elos de forma que G seja simétrico;
2. Atribuir sentido aos elos restantes;
3. Determinar a travessia atual de G.

Ford & Fulkerson propuseram o seguinte procedimento para transformar um grafo misto em grafo orientado, (Ford & Fulkerson, 1962).

Procedure *Ford-Fulkerson* (G, G');
// Descrição: Transformação de um grafo misto em um grafo orientado

Passo 1: Trocar cada elo de G com um par de arcos orientados opostos, assim obtendo um grafo orientado G' = (V, A'). Atribuir a cada arco A $\cap$ A' um limite inferior 1 e a cada arco A'-A um limite inferior 0. Atribuir também a cada arco de A' um limite superior de 1.

Passo 2: Usando um algoritmo de fluxo em redes, determinar uma circulação de fluxo viável em G. Fazer x_{ij} ser um fluxo no arco $<v_i,v_j>$.

Passo 3: Orientar alguns elos de G do seguinte modo: se $(v_i, v_j) \in$ E, x_{ij}=1 e x_{ij} = 0, orientar (v_i, v_j) de v_i a v_j.

Um procedimento para orientar completamente um grafo simétrico, pode ser colocado como segue.

Procedure *OrientandoCompletamente* (G, G^o);
// Descrição: Transformação de um grafo simétrico (euleriano) em um grafo orientado

Passo 1: Se todos os elos são orientados, parar.

Passo 2: Fazer v ser o vértice que contém pelo menos um elo (v,w). Fazer v_1=v e v_2=w.

Passo 3: Orientar (v_1, v_2) de v_1 a v_2. Se v_2=v, voltar ao passo 1.

Passo 4: Fazer v_1=v_2 e identificar um elo (v_1,v_2) incidente a v_1. Voltar ao passo 3.

Nota-se que o procedimento acima não garante a orientação de todos os elos na primeira fase (Ford-Fulkerson). Nota-se também que a solução final não necessariamente será a do carteiro misto, a menos que algumas propriedades específicas relacionadas exclusivamente à instância sejam atendidas. Porém o segundo método atende à orientação completa desde que G seja euleriano.

Teorema 1 (Ford & Fulkerson): Seja G um grafo *f*-conexo misto. G é euleriano (G_E) se e só se para cada subconjunto S de vértices de G, o número de arcos e elos de V\S (ou V-S) para S e o número de elos de S para V\S (ou V-S) é um inteiro par não negativo.

Na figura 6, mostra-se que no grafo à esquerda como a parte indicada do grafo viola as condições do teorema 1, o grafo misto não é pois euleriano. Já no grafo da direita, não há um subconjunto S que viole o teorema, por isso mesmo o grafo é euleriano. Um de seus percursos possíveis é Γ = (D-g-C-d-A-a-B-i-C-d-A-c-D-f-B-b-A-h-D). Segue daí um importante teorema proposto por Ford & Fulkerson:

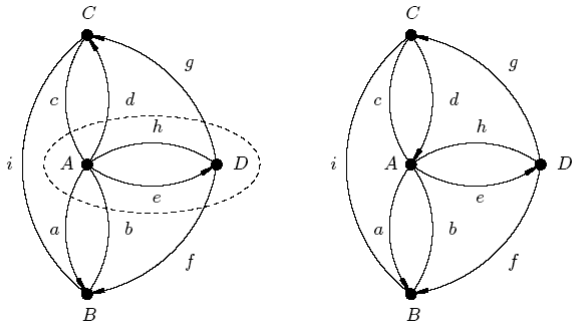

Figura 6 – A esquerda um grafo não euleriano misto e à direita um grafo euleriano misto.

Teorema 2 (Ford & Fulkerson): Seja G um grafo *f*-conexo misto. *Há um algoritmo polinomial que tanto encontra um percurso euleriano de G (se G é euleriano), tanto mostra que G não é euleriano.*

Nobert & Picard propuseram uma formulação matemática onde há apenas uma variável y_{ij}, associada a cada elo de *E*. A solução de programação inteira portanto não especifica a dimensão dos elos. São impostas restrições para que o grafo aumentado satisfaça as condições necessárias e suficientes de unicursalidade, ou seja, o grafo deve ser par e balanceado (Nobert & Picard, 1996).

Seja um subconjunto próprio e qualquer *S* de *V*, onde *V* é o conjunto de vértices, os conjuntos:

$A^+(S) = \{<v_i,v_j> \in A: v_i \in S, v_j \in V\text{-}S\}$

$A^-(S) = \{<v_i,v_j> \in A: v_i \in V\text{-}S, v_j \in S\}$

$E(S) = \{(v_i,v_j) \in E: v_i \in S, v_j \in V\text{-}S \text{ ou } v_i \in V\text{-}S, v_j \in S\}$

e seja $u(S) = |A^+(S)| - |A^-(S)| - |E(S)|$. Portanto, se $S = \{v_k\}$, então $A^+(S) = A^+(k)$, $A^-(S) = A^-(k)$, e $E(S) = E(k)$.

As constantes p_k e as variáveis z_k são definidas como seguem, e apenas uma variável y_{ij} é definida para cada (v_i, v_j) que devem ser adicionadas ao grafo para torná-lo euleriano, a qual deve ser inteira haja vista o número inteiro de passagens por este elo. A formulação é dada a seguir.

$$\text{[Misto-1] Minimizar} \sum_{<v_i,v_j> \in A} c_{ij}\, x_{ij} + \sum_{<v_i,v_j> \in E} c_{ij}\, x_{ij} \qquad [4.1]$$

s.a.

$$\sum_{<v_i,v_j> \in A^+(S)} x_{ij} + \sum_{<v_i,v_j> \in A^-(S)} y_{ij} = 2z_k + p_k \ (v_k \in V) \qquad [4.2]$$

$$-\sum_{<v_i,v_j> \in A^+(S)} x_{ij} - \sum_{<v_i,v_j> \in A^-(S)} x_{ij} + \sum_{<v_i,v_j> \in E} y_{ij} \geq u(S),\ S \subset V,\ S \neq \phi \qquad [4.3]$$

$$z_k, x_{ij}, y_{ij} \in Z_+ \qquad [4.4]$$

Nesta formulação [4.2] a [4.3] são as restrições que forçam com que todos os subconjuntos próprios não vazios *S* e *V* sejam balanceados, isto é feito pela imposição de que um número suficiente de arcos e elos sejam introduzidos para compensar a falta de balanceamento de *u(S)*. A inclusão destas restrições foi feita como uma forma generalizada para o tratamento das desigualdades de *blossons* [4.5].

$$\sum_{<v_i,v_j> \in A^+(S)} x_{ij} - \sum_{<v_i,v_j> \in A^-(S)} x_{ij} + \sum_{<v_i,v_j> \in E} y_{ij\,j} \geq 1,\ S \subset V,\ V \text{ é impar} \qquad [4.5]$$

Nobert & Picard resolveram instâncias de grafos os quais continham um número de vértices variando entre $16 \leq |V| \leq 225$, o número de arcos variando em torno de $2 \leq |A| \leq 5569$ e elos em torno de $15 \leq |E| \leq 4455$ em tempo razoável. As instâncias destes modelos não foram disponibilizadas pelos autores, porém usaram grafos grade como os aqui disponibilizados.

Seja $\vec{G} = (V, A \cup E^+ \cup E^-)$ o grafo direcionado associado a *G*. Para cada $e \in E$, faça $c_{e^+} = c_{e^-} = c_e$. Uma circulação não negativa *x* em $\vec{G}$ é o vetor de incidência da rota do carteiro se e somente se $x_a \geq 1$ para todo $a \in A$, e $x_{e^+} + x_{e^-} \geq 1$ para todo $e \in E$. Esta segunda formulação foi apresentada por Kappauf *et al.* (1979) a qual é proposta como uma formulação de programação inteira. Os mesmos autores demonstraram que o poliedro correspondente à relaxação linear desse modelo tem pontos extremos "meio inteiros" (Kappauf *et al.*, 1979). Esta formulação também foi acompanhada por Christofides *et al.* (1984), que se basearam na caracterização de Veblen dos grafos Eulerianos mistos e a formulação de fluxos similar ao do caso direcionado (Martinez, 2003):

$$\text{[PCC Misto – 2] Minimizar } \sum_{<v_i,v_j> \in A} c_{ij}x_{ij} + \sum_{(v_i,v_j) \in E^+} c_{ij}x_{e^+} + \sum_{(v_i,v_j) \in E^-} c_{ij}x_{e^-} \qquad [4.6]$$

sujeito a,

$$x(\vec{\delta}(v)) - x(\vec{\delta}(v)) = 0 \text{ , para qualquer } v \in V \qquad [4.7]$$

$$x_a \geq 1 \text{ para qualquer } a \in A \qquad [4.8]$$

$$x_{e^+} + x_{e^-} \geq 1 \text{ para qualquer } e \in E \qquad [4.9]$$

$$x_l \geq \aleph \text{ para qualquer } l \in A \cup E^+ \cup E^- \qquad [4.10]$$

Nesta formulação, na função objetivo [4.6] deseja-se minimizar a soma dos custos dos arcos atravessados, mais a soma dos elos atravessados num sentido E^+ e noutro E^-. As equações [4.7] mantêm a unicursalidade dos vértices (equilíbrio de grau de entrada e saída dos vértices). As equações [4.8] garantem que os arcos serão visitados pelo menos uma vez. A equação [4.9] diz que um elo deve ser atravessado pelo menos uma vez em um dos dois sentidos possíveis. E finalmente a equação [4.10] diz que o número de vezes que as ligações são utilizadas na solução, deve ser inteiro.

Yan & Thompson (1998) também apresentaram uma variação B&B baseada na formulação geral de Gendreau, Laporte & Zhao (1990), que na verdade atua sobre a formulação do problema do carteiro chinês íngreme (chamada de formulação genérica). A essa formulação adiciona-se um algoritmo B&B, o qual é uma modificação do algoritmo de subtração de colunas do problema de particionamento de conjuntos proposto por Harche & Thompson

(1994). A experiência computacional acontece sobre um conjunto de instâncias aleatórias, $15 \leq |V| \leq 99$ e $27 \leq |E| \leq 179$ onde se prova facilmente, para elas, a otimalidade já mesmo sobre as heurísticas discutidas no texto.

Algoritmo de Sherafat para o PCC Misto

Uma outra abordagem foi explorada por Sherafat (1988), implementada no sistema XNÊS, a qual não considera inicialmente as condições de unicursalidade do grafo. A ideia central é buscar a unicursalidade ao longo do processo de encaminhamento da solução. Sherafat aplicou um modelo de fluxo para um grafo transformado, onde os elos são convertidos em pseudo-ramos, figura 7. A direção do fluxo nos pseudo-ramos é considerada na arborescência da solução do PCC Misto, elemento que facilita o cálculo do percurso do carteiro.

No problema de fluxo a estrutura dos pseudo-ramos, figura 7, permite-se a passagem de fluxo de x_i a x_j, via os arcos $<x_j,v_r>$, $<v_r,u_r>$ e $<u_r,x_i>$ ou na direção contrária via arcos $<x_i,v_r>$, $<v_r,u_r>$ e $<u_r,x_j>$. Em ambas as direções o fluxo existirá, obrigatoriamente, passando por $<v_r, u_r>$. A este arco associa-se o custo de (x_i,x_j), os demais terão custo nulo.

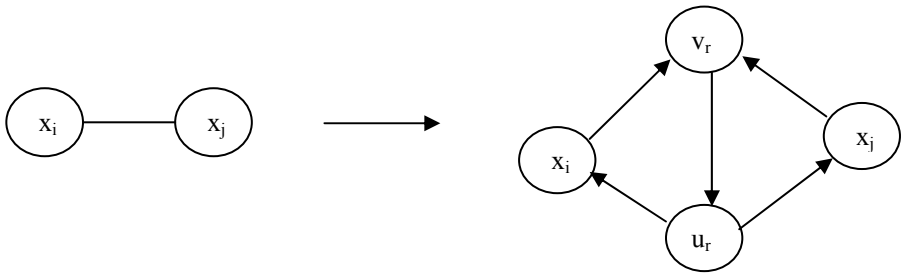

Figura 7 – Conversão de um elo em pseudo-ramos não unicursais.

Chamando de $s \in S$ os arcos dos pseudo-ramos, o grafo transformado G' terá então novos vértices, $V' = X \cup V \cup U$, e A' como um novo conjunto de arcos.

O seguinte modelo se aplica ao problema (Sherafat, 1988):

$$\text{Minimizar} \sum_{ij \in A} c_{ij} f_{ij} + \sum_{<v_i,v_j> \in S} c_{ij} f_{ij}. \qquad [4.11]$$

s.a.

$$f_{ij} \geq 1, \quad \forall < x_i, x_j > \in (A \cup S) \qquad [4.12]$$

$$\sum_{x_i \in V'} f_{ij} - \sum_{x_k \in V'} y_{ik} = 0, \quad (\forall x_j \in V') \qquad [4.13]$$

$$f_{ij} \in Z_+, \forall < x_i,x_j > \in (A \cup S) \qquad [4.14]$$

O modelo acima é uma versão do problema de fluxo em redes de custo mínimo, com limite inferior de fluxo para cada arco de no mínimo 1, o qual foi colocado anteriormente para a versão orientada do PCC. Aqui, porém, estão separadas as transformações realizadas nos elos para o caso orientado (via pseudo-ramos) os quais estão aqui representados pelo conjunto *S*. Note que y_{ik} não precisa ser inteira, haja vista a unimodularidade da matriz de restrições do modelo, forçada já em f_{ij}.

Utilizando um algoritmo de fluxo de custo mínimo tipo primal-dual simplex para redes, pex. *Out_of_Kilter* (Bazaraa *et al.*, 1990), obtém-se uma tentativa no direcionamento dos ramos de modo a se achar as réplicas de arcos e elos visando um circuito completo de custo mínimo. Na solução ótima, haverá pelo menos uma unidade de fluxo circulando em cada arco e pseudo-ramo. Em cada pseudo-ramo r_{ij}, o fluxo poderá passar de x_i a x_j, ou de x_j a x_i, ou eventualmente em ambos os sentidos. Entretanto, do modo como o modelo foi proposto, uma circulação de fluxo triangular não desejável em alguns pseudo-ramos poderá ocorrer, ver figuras 8 e 9.

Duas observações são feitas quanto às propriedades de *G'*. A primeira é relativa ao direcionamento da passagem de fluxo nos pseudo-ramos. Se num arco $<x_i, x_j> \in A$, exige-se uma unidade de fluxo, ela deverá achar um caminho Γ que ligue o vértice x_j ao x_i, para fechar o circuito. Γ poderá conter alguns pseudo-ramos, os quais serão forçosamente orientados conforme a direção do fluxo.

A segunda observação é quanto à passagem adicional de fluxo nos pseudo-ramos já orientados. Enquanto houver a possibilidade de direcionamento de outros ainda não orientados, uma vez que a simples orientação de tais ramos não acarreta em acréscimo no custo da solução, passar por eles uma unidade de fluxo. Este procedimento promove o direcionamento de uma maior quantidade de fluxo possível nos pseudo-ramos.

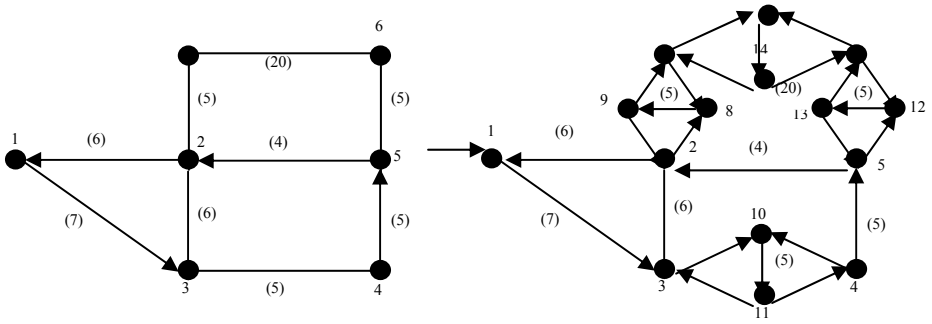

Figura 8 – Grafo onde a aplicação direta do algoritmo de custo mínimo atinge a solução do problema sem a formação de circuitos triangulares, custo do percurso ótimo = 73.

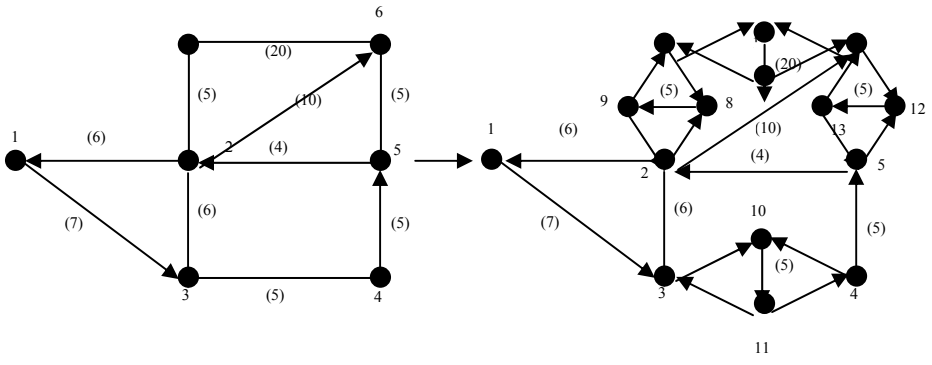

Figura 9 – Grafo onde a aplicação direta do algoritmo de custo mínimo não atinge a solução do problema sem a formação de circuitos triangulares, o fluxo triangular acontece em (6,12,13,6), ao redirecioná-lo a rota ótima é atingida após 7 iterações com custo percurso ótimo = 92.

Dado o PCC misto, se na sua solução todos os pseudo-ramos forem orientados, a solução ótima foi obtida, ou seja, as ligações do grafo *G* deverão ser multiplicadas/orientadas conforme o fluxo obtido para o grafo *G'*. Porém, se alguns pseudo-ramos permanecerem com fluxo triangular, recorre-se a uma rotina de *Branch & Bound*.

Considerando $r \in R_0$, onde R_0 representa o conjunto de todos os pseudo-ramos com fluxo triangular. Como foi visto, r corresponde a um pseudo-ramo $r_{ij} \in R$ e é constituído por um conjunto de cinco arcos (dos pseudo-ramos). Fixando a direção de fluxo neste ramo, num dos sentidos $<x_i, x_j>$ ou $<x_j, x_i>$, temos:

$$(i) \quad \begin{cases} f(x_i, v_j) \geq 1 \\ f(u_r, x_i) \geq 1 \end{cases} \qquad (ii) \quad \begin{cases} f(x_j, v_r) \geq 1 \\ f(u_r, x_i) \geq 1 \end{cases}$$

Onde $f(x_i, x_j)$ é o fluxo no ramo $<x_i, x_j>$.

Inserindo as restrições (i) e (ii) ao problema original, duas situações poderão ocorrer novamente. A primeira é quando a solução é viável para todos os pseudo-ramos, neste caso encontrou-se um limite superior do problema. A segunda deve-se a um novo R_1 de pseudo-ramos com fluxo triangular que foi obtido, para este caso tem-se que repetir o procedimento, fixando a direção de fluxo para um novo $r \in R$. Tal procedimento deverá prosseguir numa árvore até que a solução seja ótima.

Um algoritmo exato para o PCC misto considerando as colocações acima pode ser descrito como segue:

Procedimento PCC-Misto $(G, G_m, Custo)$

Dados de Entrada: Grafo (Vértices, Ligações e atributos)
Dados de Saída: Multigrafo com as réplicas das ligações e Custo Global – ou perímetro do multigrafo

Passo 0: Fazer a transformação de *G* usando pseudo-ramos.

Passo 1: Resolver o problema PCC Misto-2, P(0). Se o fluxo é viável em todos os seus pseudo-ramos a solução ótima foi encontrada. Ir ao passo 5. Caso contrário, fazer LS de custo igual a infinito. Fazer i=0 e ir ao passo 2. Fazer LI = Solução(P(0)), limite inferior (*relaxação*) da instância.

Passo 2: *Ramificação* – Escolher um pseudo-ramo r_k pertencente a R_i para direcionar. Fazer i = i+1. O novo problema a ser resolvido P(i) é constituído por P(i –1) acrescido de,

$$f(x_i, v_{r_k}) \geq 1$$
$$f(u_{r_k}, x_j) \geq 1$$

Ir ao passo 3.

Passo 3: *Limitantes* – Resolver o problema P(i). Calcular o custo de circulação (LC) de P(i).

- Se LC ≥ LS ir ao Passo 4.
- Se LC < LS e a solução corrente é viável, fazer LS = LC, armazenar o vetor de fluxo e ir ao passo 4.
- Se LC < LS mas a solução não é viável, ir ao passo 2.

Passo 4: *Backtracking* – Seja $t \in R_j$ o último pseudo-ramo orientado, mas ainda não reorientado; *j* representa um estágio em que *t* foi orientado. Redirecionar *t* no sentido contrário. Fazer i = i + 1. O novo problema P(*i*) é P(*j+1*) acrescido por,

$$f(x_i, v_{r_t}) \geq 1$$

$$f(u_{r_t}, x_j) \geq 1$$

vá ao passo 3.

(*Corte por Otimalidade*) Se não há nenhum pseudo-ramo *t* não reorientado, a solução ótima foi encontrada. Ir ao passo 5.

Passo 5: Gerar o grafo aumentante de *G* a partir da solução armazenada no último vetor de fluxo registrado (ou seja, o grafo *G* será descrito pelo número de vezes que se passa por cada arco, e o número de vezes sobre o percurso dos pseudo-ramos que descrevem cada orientação de elo). Terminar.

No algoritmo de Sherafat (1988) os pseudo-ramos que resistem ao direcionamento não são fixos, na verdade se limita ao número de problemas que serão criados no algoritmo *Branch & Bound*. Cada pseudo-ramo orientado nos passos 2 e 4 deverá fechar um novo ramo orientado em algum $r \in R_i$, e/ou duplicar outros que já têm fluxo. Pelas razões já expostas, os ramos R_i por não aumentarem o custo teriam mais do que os outros, possibilidades de atraírem esse fluxo e se autodirecionarem. Desta forma, os pseudo-ramos sem fluxo viável se esgotam rapidamente. Além do mais, a solução do problema de fluxo P(i), no passo 3 do algoritmo, geralmente requer um pouco de esforço computacional, uma vez que o problema P(i -1) é resolvido na etapa anterior. Na maioria das vezes, poucas atualizações nas variáveis primal e dual do algoritmo primal-dual simplex são feitas, tornando o processo muito rápido necessitando apenas de atualização do vetor de fluxos correntes.

O método de Sherafat, como foi projetado originalmente, faz poda na árvore B&B considerando o limite superior (LS) das soluções viáveis encontradas uma a uma. O limite inferior calculado, como ele assim o denomina originalmente, é de fato um valor de controle da solução corrente, e não propriamente o verdadeiro Limite Inferior atualizado de um processo B&B, tipo LI/LS.

Para que a busca na árvore B&B se esgote em geral mais rapidamente, é preciso que seja calculada inicialmente uma solução viável, onde as orientações de fluxo e pseudo-ramos gerados para a sua construção são guardadas como entrada da árvore B&B, e uma vez que a sua viabilidade seja garantida, o valor desta solução deve ser tomado como o Limite Superior inicial. Em seguida, deve-se utilizar uma atualização sobre os pseudo-ramos abertos que explora a cada passo as ramificações de seus pais com o menor limite inferior, e em seguida explorar a árvore B&B no modo de busca em profundidade até novo LS ser encontrado ou ser abandonado este pseudo-ramo. Este processo deve ser feito até que se esgotem os pseudo-ramos da árvore B&B. Dentre essas ideias, neste trabalho somente a primeira foi implementada (Wolsey, 1980).

Para acelerar o método acima, uma metodologia meta-heurística foi inserida de modo a encontrar uma solução inicial que sirva como LS para o problema. Trata-se então de transformar o procedimento exato de puro B&B para um método combinado GRASP+B&B.

[GRASP+B&B] Aproveitando as ideias do método de Sherafat, de forçar a orientação de fluxo, uma metodologia GRASP pode ser facilmente construída, tomando como lista de

candidatos os pseudo-ramos que geraram fluxo triangular (Feo & Resende, 1995). Podemos então tomar uma parte desta lista, e selecionar aleatoriamente um pseudo-ramo (digamos ordenados na lista em ordem decrescente de custo) de uma forma gulosa e orientá-lo em um sentido selecionado também de forma aleatória. Após isto, aplicar o algoritmo de fluxo de custo mínimo sobre a rede atual. A cada iteração de seleção uma nova lista de pseudo-ramos pode ser formada, e segue-se até que uma solução (grafo) totalmente orientada seja obtida (sem pseudo-ramos). Este processo pode ser repetido um número de vezes, até se obter uma "boa" solução, e em seguida proceder a trocas locais, conforme pode ser visto em Pearn & Chou (1999), Martinez (2003), Groves & Vuuren (2005) e Negreiros & Coelho Jr (2007).

Outra metodologia GRASP fora também sugerida por Corberán, Marti & Sanchis (2002), porém neste caso eles usam um grafo aumentado sem a presença de fluxos triangulares, e procedimentos de melhoria baseados também na orientação final de fluxo do grafo resultante. A forma como orientam o grafo colocando três arcos por cada elo, e atribuindo unidade de fluxo é bastante interessante (Corberán *et al.*, 2002).

O procedimento GRASP acima sugerido, toma pois a seguinte forma:

Procedimento PCC-GRASPMisto $(G, G_m, LS, R_i, i_{\max})$

Dados de Entrada: Grafo (Vértices, Ligações e atributos).
Dados de Saída: Multigrafo com as réplicas das ligações,
LS – Limite superior – ou perímetro do multigrafo;
R*i* – é o conjunto formado pelos pseudo-ramos da iteração i na descida da árvore de construção de uma solução;
i_{max} – é a quantidade de descidas na árvore de construção de uma solução inicial.

Passo 0: Fazer a transformação de *G* usando pseudo-ramos.

Passo 1: Resolver o problema PCC Misto-2, P(0). Se o fluxo é viável em todos os seus pseudo-ramos a solução ótima foi encontrada. Caso contrário, fazer LS de custo igual a infinito. Fazer i=0, tomar R (e R_0=R) como sendo o conjunto de pseudo-ramos ainda não orientados, fazer LS = $+\infty$, e ir ao passo 2. Fazer LI = Solução(P(0)), limite inferior (*relaxação*) da instância.

Passo 2: Fazer para um número de iterações (Estipuladas pelo usuário).

Passo 2.1: Enquanto houver pseudo-ramos a orientar ($R \neq \phi$), i:=i+1; faça.

Passo 2.1.1: Ordenar o conjunto R de pseudo-ramos na ordem decrescente dos custos do arco valorado positivamente.

Passo 2.1.2: Tomar do conjunto R, os alfa (p.ex. 15) primeiros elementos, e selecionar aleatoriamente um dos pseudo-ramos para orientar.

Passo 2.1.3: Escolher aleatoriamente uma direção de orientação do pseudo-ramo. E aplicar a orientação. Fazendo,

$$f(x_i, v_{r_k}) \geq 1$$
$$f(u_{r_k}, x_j) \geq 1$$

Passo 2.1.4: Aplicar o algoritmo de circulação, retornando o grafo de circulação final e o custo de circulação atual LC.

Passo 2.1.5: Se no grafo de circulação final (LC é o seu custo de circulação) existirem ainda pseudo-ramos não direcionados, tornar R=ϕ, e atualizar os pseudo-ramos desta circulação em R, (armazene em R_i). Voltar ao passo 2.1.2.
Caso contrário Se LS > LC então guarde esta solução de grafo aumentado, e faça LS:=LC, i_{max}:=i.

Passo 2.2: Incremente as iterações, torne R=ϕ, faça i:=0, e retorne ao passo 2.

Passo 3: Aplicar uma metodologia de melhorias sobre o valor da solução corrente (Pearn & Chou, 1999; Negreiros & Coelho Jr, 2007).

Passo 4: Gerar o grafo aumentante de *G* a partir da solução armazenada no último vetor de fluxo registrado (ou seja, o grafo *G* será descrito pelo número de vezes que se passa por cada arco, e o número de vezes sobre o percurso dos pseudo–ramos que descrevem cada orientação de elo). Terminar.

As heurísticas de melhorias propostas por Pearn & Chou, são relativas às soluções dos métodos construtivos (MIXED-1, MIXED-2 e MIXED-12/21) propostos por Frederickson (1979). No entanto, se adaptam perfeitamente ao nosso caso, pois na verdade atuam sobre a melhoria da solução de fluxo de custo mínimo de um grafo orientado euleriano. A seguir temos esta heurística de melhoria e uma nova, variante da primeira.

1. *Improved Heuristic – Pearn & Chou*

Seja Ai o conjunto dos arcos orientados (dos pseudo-ramos) e aumentados (acrescentados em fluxo) de G(V,E,A), formando o grafo direcionado D(V, $A \cup A_i$):

(Passo 1): Aplique o algoritmo de fluxo de custo mínimo (OOK) sobre a rede D sem os arcos aumentados e apenas uma cópia no sentido de cada pseudo-ramo orientado D(V,A'), onde se tem os novos arcos Ad.

(Passo 2): Troque os arcos aumentados Ad por Ai para obter a nova solução, DD(V,$A \cup A_d$). Note que para aqueles elos contendo duas direções diferentes em D, cada direção deverá ser escolhida para gerar uma solução completa, e a melhor das duas é selecionada.

2. *Generic Improvement Heuristic – Negreiros & Coelho Jr*

Seja Ai o conjunto dos arcos orientados (dos pseudo-ramos) e aumentados (acrescentados em fluxo) de G(V,E,A), formando o grafo direcionado D(V,$A \cup Ai$), solução da fase aleatória do GRASP, ou mesmo após se encontrar um LS no procedimento de Sherafat (Negreiros & Coelho Jr., 2007). Este método de melhoria é uma generalização do método sugerido por Pearn & Chou.

Tome em D agora o seguinte:

1. Para os Arcos: Coloque como limite inferior de fluxo 1, e limite superior de fluxo +∞.

2. Para os Pseudo-Ramos:

2.1. Sendo o pseudo-ramo orientado num único sentido: Manter o limite inferior como 1 e o limite superior como sendo um valor +∞;

2.2. Sendo o pseudo-ramo orientado em dois sentidos: tomando o sentido com menor número de cópias, o limite inferior de fluxo será zero e no outro será +∞. No outro sentido (Maior número de cópias), colocar como limite inferior de fluxo 1 e superior +∞.

3. Aplique o algoritmo OOK sobre D(V,A'), retornando o multigrafo DM. A solução de fluxo neste grafo será euleriana e sempre menor ou igual à solução do grafo orientado anterior.

Um detalhe importante sobre o método acima, é que a orientação de D é ótima com respeito a este grafo. Ou seja, DM é resultante de uma orientação ótima sobre D, e obviamente sobre este grafo orientado a solução de fluxo de custo mínimo, ou do carteiro orientado também é ótima (Negreiros & Coelho Jr, 2007). Para tornar o método B&B mais atrativo, pode-se então incluir este último método de melhoria toda vez que se atingir uma solução viável (LS). Trata-se aqui de uma *hibridização* do método B&B proposto anteriormente, uma vez que se introduz uma nova propriedade de busca à medida que se chega a uma solução viável. Tomando-se a orientação de menor custo do grafo reduzido e consequentemente o multigrafo resultante da aplicação do fluxo, a árvore B&B será bastante reduzida e o desempenho do método bastante melhorado, quando finalmente altera-se o teste de terminação, avaliando se LS-LI ≤ Custo da menor ligação do grafo.

O algoritmo exato geral **Combinado&Híbrido** seria então:

Procedimento Combinado&Hibrido(G, G', LS);

Dados de Entrada: Grafo (Vértices, Ligações e atributos);
Dados de Saída: Multigrafo com as réplicas das ligações,
LS – Limite superior – ou perímetro do multigrafo;
Ri – é o conjunto formado pelos pseudo-ramos de cada iteração i (i=1 a $i_{\max}$) na descida da árvore de construção de uma solução;
$i_{\max}$ – é a quantidade de descidas na árvore de construção de uma solução inicial.

begin
PCC-GRASPMisto $(G, G_m, LS, R_i, i_{\max})$ **;**
PCC-Misto $(G, G_m, R_i, i_{\max}, LS)$; **(*)**
end; // Combinado&Hibrido

(*) Esta chamada introduz no método PCC-Misto os conjuntos dos pseudo-ramos abertos a cada nível (R_i) da árvore B&B cuja orientação de descida é assumida pela solução final GRASP, os parâmetros R e i_{max} devem ser tomados de tal modo que a primeira descida recursiva na árvore B&B deve ser feita através deste caminho, seguindo daí para suas ramificações já definidas.

Para o método original de Sherafat, o autor relata que em suas experiências computacionais as soluções se mostraram muito demoradas à medida que o número de elos aumenta proporcionalmente aos arcos. É de se esperar que esta técnica se torne atrativa quando se tem um grafo onde a dominância é maior no número de arcos, porém isto pode ou não ser sempre verdade, como é demonstrado pelos resultados dos testes da próxima seção.

É importante reforçar aqui que, na forma como é resolvido o PCC-Misto, tem-se uma aproximação a um processo de puro direcionamento de um grafo. Isto significa que quanto maior a parte da rede com elos, pior será o desempenho deste método. Assim, para grafos puramente simétricos, recomenda-se que se use o método simétrico e não este.

5. Avaliação Computacional dos Métodos

O método simétrico já foi bastante explorado pela literatura, tendo recebido várias adaptações principalmente no que concerne à estrutura de dados e a forma de ataque ao método proposto por Edmonds & Johnson. Dentre os principais autores Derigs propôs algumas estratégias de baixa complexidade que dizem sobre como realizar o caminho aumentante, assim como Gabow utilizou estruturas de listas de adjacências, propostas por Tarjan que se mostraram muito eficientes, tornando esta classe de métodos com complexidade $O(|V||E|.\alpha(|E||V|))$, onde os anteriores eram $O(|V|^4)$, como proposto por Edmonds & Johnson. Mais adiante, Heske propôs uma modificação no método de rotulação de Gabow, tendo conduzido a solução para $O(|V||E|)$. O método proposto por Gabow, teve grande mérito, e por isso mesmo incorporou uma redução de complexidade na resolução do PCC simétrico de forma expressiva. A implementação que fizemos, foi uma adaptação daquela proposta por Bukard & Derigs, muito conhecida e superior às anteriores citadas conforme o autor. Foi transcodificada de FORTRAN IV para ANSI-C no ambiente Borland C++ (Borland Internacional) (Derigs, 1988; Gabow, 1975; Bukard & Derigs, 1980; Edmonds & Johnson, 1973; Heske, 1978).

Já para o caso do método para a versão mista, o algoritmo B&B proposto por Sherafat foi implementado, porém usando Visual C++, sobre o ambiente Visual Studio (Microsoft). A base do método de Sherafat é o algoritmo de *Out-of-Kilter*, proposto em Bazarra & Jarvis (1990), implementado neste código, e que também pode ser visto em muitos livros e artigos de otimização de fluxo em redes como Ahuja *et al.* (1993).

Foram testadas ao todo 45 instâncias divididas entre simétricas e mistas:

- Instâncias Simétricas (13): Aldeota, (Aldeota+Meireles+Iracema+Centro) e Centro de Fortaleza, Penha (Vitória/ES), Derigs, grafos grade com densidade definida por vértice (GR1222, GR1020_20, GR1010), e 5 instâncias randômicas;
- Instâncias Mistas (32): compostas por 20 geradas randomicamente; centro da cidade de Belo Horizonte, Bairro Penha de Vitória/ES, setores da coleta de lixo do Rio de Janeiro (Botafogo, Copacabana e Urca): BA210, BA229, BA237, e da coleta de lixo de Fortaleza C476 e C474 (Meireles-Ideal, Praça Portugal-Náutico), grafos do controle do dengue – percurso do fumacê: Papicu, Quintino Cunha, Barra do Ceará e Centro, e Corberán versão com custos euclidianos. As 20 instâncias geradas randomicamente têm $100 \le |V| \le 500$, $112 \le |E| \le 584$ e $0 \le |A| \le 400$. Estas instâncias podem ser divididas em cinco grupos (100, 200, 300, 400 e 500 vértices) com quatro ou cinco instâncias em cada um. Cada grupo possui uma quantidade constante de elos e de vértices; aumentando gradualmente a quantidade de arcos.

As figuras 10 a 13 apresentam a visualização de três instâncias diferentes deste conjunto utilizando o ambiente gráfico do Sistema XNÊS. O grupo das 19 instâncias foi propositadamente feito usando grafos planares, caracterizados como "*grafos reais*" (Grotschel & Win, 1993), pois mantêm a estrutura de cruzamentos de ruas e distâncias entre cruzamentos urbanos comuns às cidades. Todas as instâncias estão disponíveis em www.lcc.uece.br/~negreiro ou diretamente com o primeiro autor deste artigo.

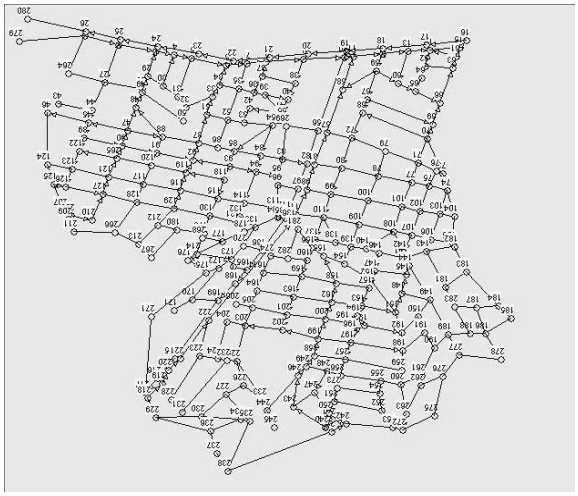

Figura 10 – Grafo representando o centro da cidade de BH.

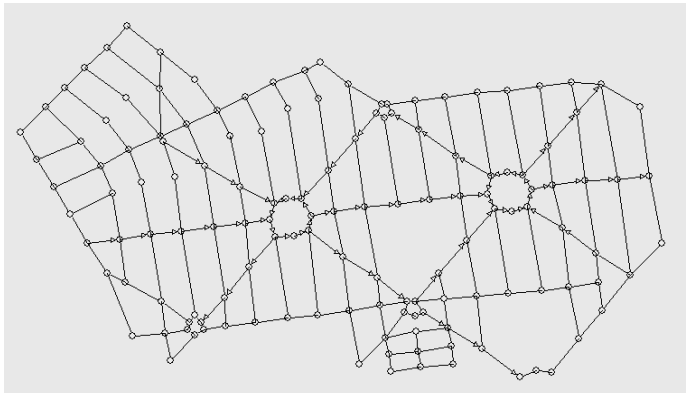

Figura 11 – Visualização gráfica de uma instância da cidade de Vitória/ES – MPenha.GRF.

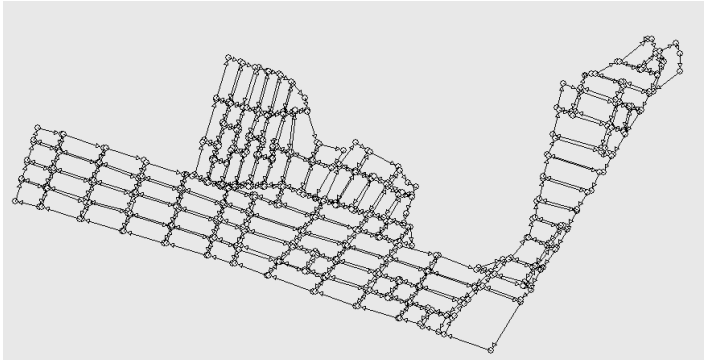

Figura 12 – Visualização gráfica de uma instância PapicuB.GRF, controle da dengue – Veículo Fumacê.

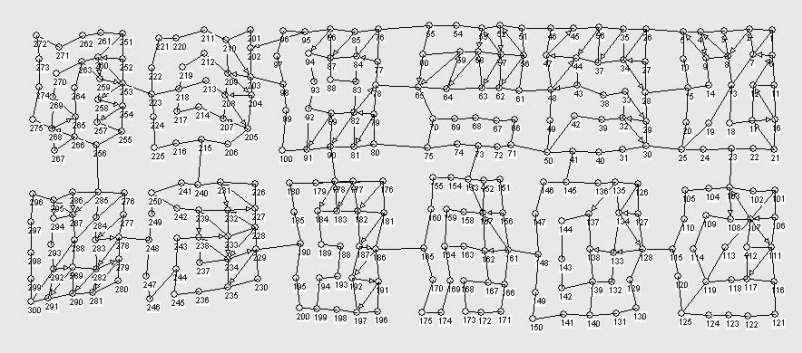

Figura 13 – Visualização gráfica de uma instância randômica com 300 vértices.

Em todos os testes realizados para o caso misto, executou-se inicialmente a meta-heurística GRASP usando o método de melhoria genérico (Negreiros & Coelho Jr., 2007). Com uma solução inicial de "boa-qualidade", a utilizamos como limite superior inicial no algoritmo B&B, como descrito no método exato **Combinado&Híbrido**.

As instâncias para avaliar o algoritmo proposto, foram elaboradas considerando o custo unitário para toda ligação do grafo. E o custo real das distâncias entre vértices. Conforme Martinez (2003), a avaliação de instâncias de natureza unitária (ligações com custo unitário) são de grande valia com relação aos tempos necessários para se encontrar a otimalidade e a distância do custo relativamente ao perímetro total do grafo. Acrescentamos a isto, que estas instâncias também servem para avaliar bem os percursos construídos pelos métodos, haja vista a ausência de diferença de custo entre as ligações, e o percurso no grafo está associado exclusivamente a sua topologia.

Para as instâncias de grafos reais, foi dado um limite de tempo para sua execução de até 5h (18000 segundos), onde variamos de 1 em 1 hora os tempos para os grupos de instâncias.

Na fase GRASP, o valor do percentual do tamanho da lista (alfa) de pseudo-ramos com fluxo triangular foi de α=20%. Já na fase B&B, além do critério de limite, outro critério de parada introduzido foi a tolerância da diferença entre limites (LS-LI $\leq \min\{c_{ij}, \forall i,j \in E \cup A\}$).

Foram utilizadas seis máquinas diferentes para a realização dos testes, a tabela 1, apresenta as configurações dos respectivos equipamentos:

Tabela 1 – Características dos equipamentos utilizados para os testes.

Máquina	Nome	Configuração
1	Monet	Pentium 4 3.06Ghz 512Mb Ram 160Gb HD
2	DaVinci	Celeron D 2.66Ghz 512Mb Ram 60Gb HD
3	Portinari	Celeron D 2.66Ghz 256Mb Ram 80Gb HD
4	Kokopelle	Pentium 4 1.7Ghz 768Mb Ram 60Gb HD
5	Michelangelo	AMD Athlon XP 2200+ 1.6Ghz 1Gb Ram 60Gb HD
6	Renoir	AMD Athlon XP 1800+ 1.16Ghz 768Mb Ram 40Gb HD

5.1 Experimentos e Resultados Computacionais

5.1.1 Caso Simétrico

As tabelas 2 e 3 apresentam os resultados obtidos com as instâncias simétricas (caso unitário e custo positivo) usando a máquina 6 descrita na tabela 1. Nessas tabelas, |V| se refere ao número de vértices do grafo, |I| se refere ao número de vértices com grau ímpar, |E| se refere ao número de elos do grafo, **Perímetro** a soma dos custos as ligações do grafo, S* ao custo da solução ótima encontrada, e t(s) o tempo do algoritmo em segundos. Os grafos estão dispostos na ordem crescente do seu número de vértices ímpares. Os tempos foram medidos após se disparar a chamada do método externamente à interface (XNÊS), ou seja, o tempo total de processamento consta da soma dos tempos da leitura do arquivo de entrada (em .TXT), o tempo do algoritmo de processar a entrada e o tempo de escrever a saída (em .TXT).

Tabela 2 – Grafos simétricos com custos unitários.

Instância	|V|	|E|	|I|	Perímetro	S*	t(s)
100_0.GRF	100	112	20	112	137	0.071
200_0.GRF	200	240	46	240	284	0.080
300_0.GRF	300	352	74	352	423	0.100
400_0.GRF	400	471	92	471	560	0.181
500_0.GRF	500	583	114	583	699	0.220

Tabela 3 – Grafos simétricos com custos não unitários.

GRAFO	|V|	|E|	|I|	Perímetro	S*	t(s)
GR1010.GRF	100	1001	2	263396	263397	0.080
GR1222.GRF	100	488	8	12395	12490	0.270
DERIGS.GRF	51	77	40	589	750	0.230
GR1020_20.GRF	100	1182	48	310820	313369	0.080
PENHA.GRF	140	255	62	11807	13471	0.080
ALDEOTA.GRF	414	731	168	74536	83936	0.120
CENTRO.GRF	522	823	278	79810	94374	0.190
5BAIRROSFOR.GRF	1300	2174	648	211735	243387	0.381
100_0.GRF	100	112	20	41904	50224	0.040
200_0.GRF	200	240	46	79710	94390	0.050
300_0.GRF	300	352	74	112437	135340	0.060
400_0.GRF	400	471	92	149845	179327	0.101
500_0.GRF	500	583	114	182651	219985	0.110

5.1.2 Caso Misto

Os experimentos computacionais foram projetados para avaliar a eficiência do método desenvolvido. As avaliações foram feitas sobre grafos planares e urbanos, e grafos genéricos do tipo aleatórios em grade.

5.1.2.1 Testes com Grafos Planares e Urbanos

Utilizamos duas tabelas para os testes das instâncias unitárias e mais duas para as de custo positivo (custo das ligações medido pela distância euclidiana entre vértices). Os campos utilizados na primeira tabela (X.1) de resultados são os seguintes:

Instância	Nome do grafo utilizado;
$\|V\|$	Quantidade de vértices do grafo;
$\|E\|$	Quantidade de elos do grafo;
$\|A\|$	Quantidade de arcos do grafo;
FT	Número de fluxos triangulares iniciais;
LI	Valor da solução inicial relaxada;
LH	Valor da solução da meta-heurística GRASP;
LS	Valor da solução ótima ou do último limite superior encontrado antes do término do tempo de execução, ou seja, a melhor solução viável encontrada até o momento;
GAP LS-LI	Porcentagem (%) indicando a distância que o limite superior estava do limite inferior no momento do término do tempo de execução;
GAP LS-LH	Porcentagem (%) que indica a distância das soluções encontrada pela heurística e da melhor solução encontrada até o momento;
Máquina	Máquina que foi utilizada para resolver a instância.

a segunda tabela (X.2), refere-se aos tempos de processamento, utiliza os campos:

Instância	Nome do grafo utilizado;
Th	Tempo total da meta-heurística;
T*	Tempo para se encontrar a melhor solução;
Tpo	Tempo para provar que a melhor solução encontrada também é a ótima;
Tout	Tempo total do algoritmo *out-of-kilter* incluindo o tempo que este foi processado na fase heurística mais o tempo da fase exata;
Tmediaout	Tempo médio do algoritmo *out-of-kilter*.

Para medir a qualidade dos resultados do método proposto, avaliamos o desvio para do limite inferior em relação ao limite superior assim como o desvio para o limite heurístico para o limite superior. O GAP LS-LI foi então calculado da seguinte maneira: 100 * (LS-LI)/LI e o GAP LS-LH: 100 * (LH-LS)/LS. Nas instâncias onde existe um asterisco (*), foi provada a otimalidade, ou considera-se ótima a solução corrente de forma incumbente (por diferença de limites).

O critério de otimalidade, considerando os exemplos dados, tem haver também com o rigor da distância entre LS e LI. Durante os testes, nós consideramos a parada do B&B apenas nas condições propostas por Sherafat. No entanto, diante dos percentuais, e da integralidade das soluções, todas as instâncias que atingiram valores LS-LI inferior ou igual ao custo da menor ligação da rede, em relação ao primeiro limite inferior do modelo relaxado (primeira orientação do grafo), foram (e podem ser) consideradas de valor ótimo.

Todos os custos das instâncias nas versões urbanas, com custo diferente de 1, foram considerados como valores inteiros positivos, haja vista o método *Out-of-Kilter* programado, proposto por Bazaara & Jarvis (1980). Note-se que se um método inteiro se aplica a uma instância com dados em reais, podemos torná-la inteira o quanto desejarmos ter em precisão na saída. Uma vez que os limites de fluxo deste problema são sempre inteiros, a matriz de fluxo será unimodular, e o fluxo também será inteiro (Ahuja *et al.*, 1993). O sistema XNÊS possui um multiplicador de escala, os elos/arcos podem ser reconfigurados diretamente aplicando um fator de multiplicação, o qual é transferido daí para o módulo de resolução do algoritmo pertinente (Simétrico, Orientado ou Misto).

Tabela 4.1a – Avaliação de qualidade de solução do método combinado GRASP-B&B em grafos com custos unitários.

Instância	|V|	|E|	|A|	FT	LI	LH	LS	GAP LS-LI	GAP LS-LH	Má-quina
100_10 Unitario.grf	100	112	10	54	150	156	151	0.67	3.31	3
100_50 Unitario.grf	100	112	50	57	197	199	199	1.02	0.00	3
100_80 Unitario.grf	100	112	80	57	227	238	230	1.32	3.48	3
100_100 Unitario.grf	100	112	100	43	258	262	259	0.39	1.16	3
200_20 Unitario.grf	200	240	20	182	326	353	330	1.23	6.97	2
200_50 Unitario.grf	200	240	50	153	359	376	360	0.28	4.44	2
200_100 Unitario.grf	200	240	100	133	406	419	408	0.49	2.70	2
200_150 Unitario.grf	200	240	150	84	469	470	470	0.00	0.00	6
300_30 Unitario.grf	300	352	30	319	471	495	472	0.21	4.87	6
300_70 Unitario.grf	300	352	70	271	531	532	532	0.00	0.00	6
300_120 Unitario.grf	300	352	120	219	578	579	579	0.00	0.00	6
300_200 Unitario.grf	300	352	200	166	667	684	669	0.30	2.24	2
400_50 Unitario.grf	400	471	50	424	676	677	677	0.00	0.00	1
400_100 Unitario.grf	400	471	100	362	737	738	738	0.00	0.00	4
400_200 Unitario.grf	400	471	200	276	856	860	860	0.47	0.47	1
400_300 Unitario.grf	400	471	300	245	924	925	925	0.00	0.00	4
500_100 Unitario.grf	500	584	100	448	893	895	895	0.22	0.00	1
500_200 Unitario.grf	500	584	200	391	990	991	991	0.00	0.00	1
500_300 Unitario.grf	500	584	300	321	1114	1115	1115	0.00	0.00	2
500_400 Unitario.grf	500	584	400	281	1203	1204	1204	0.00	0.00	1
Média								**0.39**	**1.46**	

Tabela 4.1b – Avaliação de qualidade de solução do método combinado GRASP-B&B em grafos com custos unitários, em grafos reais urbanos.

Instância	|V|	|E|	|A|	FT	LI^i	LH	LS	GAP LS-LI^i	GAP LH-LS	Máq
BH.GRF	283	534	185	373	903	951	905	0.22	5.08	3
DPapicuB.GRF	524	870	486	797	1406	1703	1407	0.07	21.17	3
DCentro.GRF	1066	973	991	894	2176	2409	2177	0.04	9.67	6
DBarradoCeara.GRF	1383	2486	1179	2226	3952	4849	3953	0.02	22.69	6
Média								**0.09**	**14.86**	

Tabela 4.2a – Avaliação de tempo de solução do método combinado GRASP-B&B em grafos com custos unitários.

Instância	Th (s)	T* (s)	Tpo (s)	Tout (s)	Tmediaout (s)
100_10 Unitário.grf	1,204	313,125	-	3583,086	0,00700202
100_50 Unitário.grf	1,047	1845,969	-	3580,267	0,00621262
100_80 Unitário.grf	1,609	81538,090	-	204881,200	0,00903530
100_100 Unitário.grf	1,312	2396,110	-	3582,901	0,00746607
200_20 Unitário.grf	5,359	1,000	-	3585,469	0,01775653
200_50 Unitário.grf	5,125	37,531	-	3585,471	0,01856449
200_100 Unitário.grf	5,234	1076,453	-	3581,445	0,01567839
200_150 Unitário.grf	4,688	1134,328	-	3583,585	0,01920968
300_30 Unitário.grf	12,485	551,015	-	3588,95	0,04408650
300_70 Unitário.grf	12,516	1121,672	-	3589,089	0,04048286
300_120 Unitário.grf	12,375	475,500	-	3589,458	0,04162414
300_200 Unitário.grf	16,906	2490,985	-	3586,911	0,04690796
400_50 Unitário.grf	33,359	2576,609	-	3589,776	0,08173442
400_100 Unitário.grf	32,562	219,531	-	3587,277	0,06520188
400_200 Unitário.grf	32,812	77,047	-	3588,475	0,06999999
400_300 Unitário.grf	31,047	31,047	-	3586,932	0,07308337
500_100 Unitário.grf	50,25	1997,812	-	3589,555	0,12606873
500_200 Unitário.grf	46,766	509,187	-	3582,564	0,07166561
500_300 Unitário.grf	48,812	2121,297	-	3586,696	0,10621897
500_400 Unitário.grf	44,875	1914,703	-	3584,143	0,09474090

Tabela 4.2b – Avaliação de tempo de solução do método combinado GRASP-B&B em grafos com custos unitários.

Instância	Th (s)	T* (s)	Tpo (s)	Tout (s)	Tmediaout (s)
BH.GRF	18.437	8.188	-	10747.94	0,07582697
DPapicuB.GRF	27.328	11.281	-	17749.86	0,08078765
DCentro.GRF	16.985	6.047	-	17915.42	0.06394688
DBarradoCeara.GRF	91.390	36.813	-	17911.43	0.16045209

Tabela 5.1a – Avaliação da qualidade de solução do método combinado&híbrido GRASP-B&B em grafos planares com custos euclidianos.

ALEATÓRIA PLANAR										
Instância	**|V|**	**|E|**	**|A|**	**FT**	**LI^i**	**LH**	**LS**	**GAP LS-LI^i**	**GAP LH-LI^i**	**Máq**
100_10 Euclidiano.grf	100	112	10	54	28210	31199	28330	0,42v	10,59	3
100_50 Euclidiano.grf	100	112	50	57	36697	40527	37065	1,00v	10,43	3
100_80 Euclidiano.grf	100	112	80	57	44926	46441	44952	0,05*	3,37	3
100_100 Euclidiano.grf	100	112	100	43	49968	51568	50075	0,21v	3,20	3
200_20 Euclidiano.grf	200	240	20	182	54621	54695	54695	0,13*	0,13	2
200_50 Euclidiano.grf	200	240	50	153	61938	62025	62025	0,14*	0,14	2
200_100 Euclidiano.grf	200	240	100	133	70588	70742	70742	0,21*	0,21	2
200_150 Euclidiano.grf	200	240	150	81	79492	82550	79523	0,03*	3,84	2
300_30 Euclidiano.grf	300	352	30	319	76941	76957	76957	0,02*	0,02	6
300_70 Euclidiano.grf	300	352	70	271	86336	86395	86395	0,06*	0,06	6
300_120 Euclidiano.grf	300	352	120	216	96720	96890	96890	0,17*	0,17	6
300_200Euclidiano.grf	300	352	200	163	113296	113340	113340	0,03*	0,03	2
400_50 Euclidiano.grf	400	471	50	424	107838	107866	107866	0,02*	0,02	1
400_100 Euclidiano.grf	400	471	100	362	117660	117746	117746	0,07v	0,07	4
400_200 Euclidiano.grf	400	471	200	276	138081	138231	138231	0,10*	0,10	1
400_300 Euclidiano.grf	400	471	300	250	155091	155219	155219	0,08v	0,08	4
500_100 Euclidiano.grf	500	584	100	448	138718	138738	138738	0,01*	0,01	1
500_200 Euclidiano.grf	500	584	200	385	160505	160561	160561	0,03*	0,03	1
500_300 Euclidiano.grf	500	584	300	323	175986	176050	176050	0,03*	0,03	2
500_400 Euclidiano.grf	500	584	400	276	199402	199632	199632	0,11v	0,11	1

Tabela 5.1b – Avaliação da qualidade de solução do método combinado GRASP-B&B em grafos urbanos com custos euclidianos.

Instância	|V|	|E|	|A|	FT	LI^i	LH	LS	GAP LS-LI^i	GAP LH-LS	Máq
Corberaneuclidiano.grf	30	28	21	8	5300	6157	6052	12,42*	1,73	6
Corberan	30	28	21	8	1049	1169	1168	11,34*	0,01	7
COLETA DE LIXO										
BA210.GRF	63	18	72	8	73253	75133	75004	2,39*	0,17	7
BA229.GRF	50	14	54	9	5247	5481	5353	1,98*	2,39	7
BA237.GRF	70	48	74	28	11531	12261	12207	5,86*	0,00	7
C474.GRF	90	68	82	50	10434	12353	11801	13,10v	1,23	7
C476.GRF	93	23	133	15	17976	19589	18783	4,48*	4,29	7
MPenha.GRF	140	167	58	97	12133	16493	13917	14,70v	18,51	7
BH.GRF	283	534	185	373	33298	33383	333383	0,00*	0,00	7
DENGUE										
DPapicuB.GRF	524	870	486	797	60188	60241	60241	0,02*	0,09	7
DQuintinoCunha.GRF	534	461	502	416	59309	61016	60377	1,08*	1,04	7
DCentro.GRF	1066	973	991	894	102662	107249	104845	2,12v	4,46	7
DBarradoCeara.GRF	1383	2486	1179	2226	98569	98787	98787	0,00*	0,22	7
Média								**1.12**	**0.56**	

Tabela 5.2 – Avaliação do tempo de solução do método combinado GRASP-B&B em grafos gerais com custos euclidianos.

Instância	Th (s)	T* (s)	Tpo (s)	Tout (s)	Tmediaout (s)
ALEATÓRIA PLANAR					
100_10 Euclidiano.grf	1,688	2250,015	-	3595,854	0,02459898
100_50 Euclidiano.grf	2,063	1858,406	-	3596,867	0,02994245
100_80 Euclidiano.grf	1,734	3552,000	-	3595,858	0,02890120
100_100 Euclidiano.grf	2,125	0,890	-	3595,812	0,02905636
200_20 Euclidiano.grf	34	53,172	-	7194,31	0,09313987
200_50 Euclidiano.grf	32,64	2679,469	-	7195,123	0,11312909
200_100 Euclidiano.grf	40,422	6086,625	-	7193,701	0,09656364
200_150 Euclidiano.grf	39,782	3578,843	-	7193,975	0,11903263
300_30 Euclidiano.grf	144,484	4173,485	-	7193,859	0,33960529
300_70 Euclidiano.grf	136	5512,844	-	7194,334	0,37015507
300_120 Euclidiano.grf	170,062	284,5	-	7195,241	0,38524609
300_200 Euclidiano.grf					

400_50 Euclidiano.grf	193,344	193,344	-	7195,342	0,37085568
400_100Euclidiano.grf					
400_200 Euclidiano.grf	433,219	433,219	-	7193,446	0,76477204
400_300 Euclidiano.grf	389,016	1570,844	-	7190,514	0,75026233
500_100 Euclidiano.grf	314,11	314,11	-	7192,165	0,26937957
500_200 Euclidiano.grf	329,421	329,421	-	7193,195	0,41940382
500_300 Euclidiano.grf	361,109	1054,234	-	7184,616	0,35542773
500_400 Euclidiano.grf	299,734	299,734	-	7184,202	0,38373049
URBANAS					
Corberaneuclidiano.grf	0,047	0,031	0,922	0,937	0,00187776
Corberan.grf	0,031	0,016	0,703	0,750	0,00188442
COLETA DE LIXO					
BA210.GRF	0,297	0,297	0,234	0,625	0,00892857
BA229.GRF	0,062	0,047	0,094	0,172	0,00382222
BA237.GRF	0,141	0,078	59,087	58,930	0,00359285
C474.GRF	1,000	206,515	-	3593,621	0,02040312
C476.GRF	0,203	8,484	20,468	20,657	0,00331042
MPenha.GRF	9,043	9,043	-	7195,816	0,11108597
BH.GRF	16,11	16,11	-	10794,76	0,16228323
DENGUE					
DPapicuB.GRF	3,594	1,375	-	17945,13	0,04448052
DQuintinoCunha.GRF	4,828	4,828	-	17960,04	0,06211689

5.1.2.2 Testes com Grafos Aleatórios Grade

Avaliamos também o comportamento dos algoritmos em relação ao parâmetro GAP LS*-LI, no que se refere ao desvio entre o limite inferior e o limite superior no momento em que o algoritmo terminou (em tempo "prematuro" fixo). Aqui consideramos instâncias de grande porte com a mesma estrutura geral de construção, sobre grafos grade assimétricos não necessariamente planares (Pseudo-Manhattan).

Desenvolvemos para isso, um conjunto de instâncias, onde cada uma possui densidade controlada de elos por vértice, e sempre se garante a *f*-conectividade dos mesmos. Tais grafos randômicos são chamados "grade" (pseudo-Manhattan), ou seja, na sua construção, os vértices estão dispostos como numa grade (figura 14). No algoritmo de construção do grafo "grade", elos (>0) são inicialmente incluídos em cada um de seus vértices de tal modo que o grau de cada vértice seja o mesmo. Em seguida se incluem aleatoriamente os arcos, de acordo com a quantidade desejada em relação aos elos. Os custos são sempre positivos e escolhidos dentro de uma faixa de valores específica, pex. [1,100].

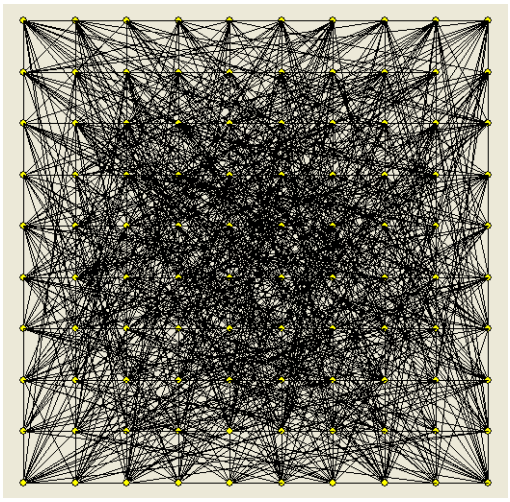

Figura 14 – Visualização de um grafo randômico tipo grade (|V|=100, |E|=100, |A|=0).

Um número de avaliações foi realizado sobre estes grafos, que basicamente nos deu o comportamento do algoritmo combinado GRASP+B&B, numa perspectiva generalista. Os gráficos das figuras 15, 16 e 17 contêm as instâncias com a mesma quantidade de vértices e de elos, mas com quantidades diferentes de arcos, usando os grafos aleatórios grade.

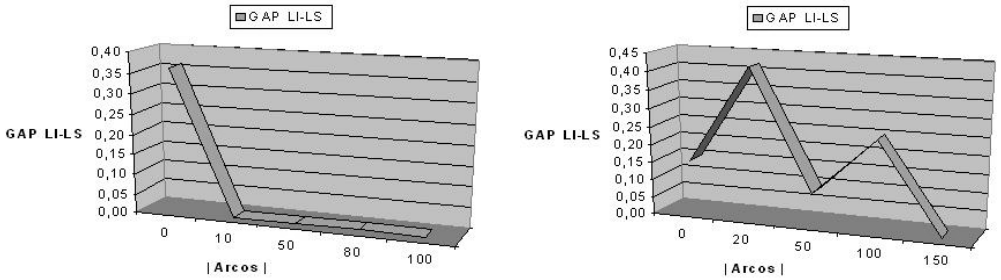

Figura 15 – Gráficos referentes ao GAP LS-LI, para os grafos no gráfico da esquerda com 100 vértices e 112 elos e os da direita com 200 vértices e 240 elos.

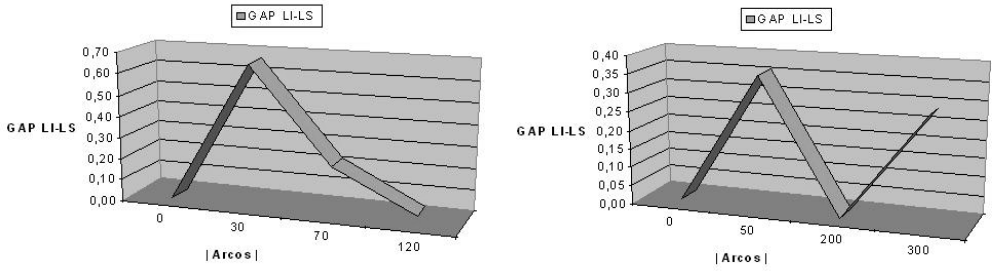

Figura 16 – Gráficos referentes ao GAP LS-LI, para os grafos no gráfico da esquerda com 300 vértices e 352 elos e os da direita com 400 vértices e 471 elos.

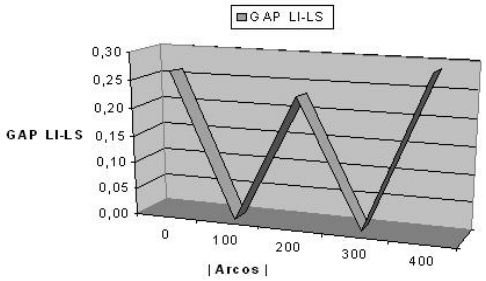

Figura 17 – Gráfico referente ao GAP LS-LI para os grafos com 500 vértices e 571 elos.

5.2 Análise Geral dos Resultados

Para os exemplos do caso simétrico, o tempo computacional do método de Busaker & Derigs é, como foi visto, proporcional ao número de vértices (de grau ímpar), e ao número de elos do grafo. A tabela 3 mostra curiosamente que o algoritmo programado não consumiu mais de 0,5 segundo em todos os 13 ensaios sobre as instâncias de pequeno, médio e grande porte. Acrescente-se o pequeno aumento no tempo com o número de elos, caracterizando a componente logarítmica de suavização do procedimento sobre o número de elos da rede. Importante dizer que neste método, se a rede já for euleriana, o mesmo não gera uma solução de caminho euleriano.

Já nos resultados dos testes das instâncias mistas unitárias podemos verificar o desempenho do método combinado que em apenas 3 das 20 instâncias a otimalidade (incumbente) não veio no tempo máximo definido.

Já para as instâncias euclidianas, encontrar mesmo o ótimo incumbente se configurou numa tarefa muito difícil para os casos C474 (lixo) e BPapicu (Dengue). A diferença entre os limites inferior e superior de todas as outras instâncias foram muito baixas com uma média 0,47% sendo que as maiores diferenças entre LS e LI, não passaram de 1%; o que demonstra ser um bom resultado para estes grafos.

Das instâncias da tabela 4.1.a (planares de custo unitário nas ligações), o custo unitário deveria "atrapalhar" o método guloso de orientação da rede. Nesses casos, a fase de melhoria exata atuou fortemente sobre a solução não permitindo que se afastasse por demais do limite superior (LS). Mesmo assim, em algumas instâncias, a fase de orientação gulosa chegou a atingir 7% além do melhor LS. Mesmo com esta discrepância, o método guloso poucas vezes ficou longe do ótimo.

Das instâncias da tabela 4.1.b (euclidianas), temos uma melhor visão sobre o impacto provocado pela orientação gulosa, sobre a presença de custo variado nas ligações. Nesses casos, 27 dos 33 resultados atingiram a otimalidade global e/ou incumbente, os demais não passaram de 1% do LS.

Na tabela 5.1.a nota-se a sensibilidade aproximadamente linear do tempo da meta-heurística ao aumento do número de vértices da rede. No entanto, o aumento do número de arcos dentro do mesmo conjunto de instâncias, mostra-se favorável quanto à obtenção da solução em termos de qualidade e tempo. As instâncias, 100_10, 100_80, 300_30 e 300_200 são aquelas onde os resultados da meta-heurística mais se afastaram do LS, e demandaram mais tempo para pequenas melhorias. Apesar disso, uma instância com poucos arcos e muitos elos

como a 200_20, pouco demandou em tempo para atingir a solução ótima, mesmo estando distante dela no valor inicial. Isto se deve à forma como estão orientados os arcos, e pequenas mudanças no fluxo dos pseudo-ramos são suficientes para que se atinja rapidamente a otimalidade. As primeiras instâncias indicadas são mais "difíceis", mesmo se investigadas manualmente.

Tempo de execução do método *out-of-kilter* foi pouco significativo, tendo acompanhado o tamanho da rede em complexidade quase-linear, no entanto mais de 99% do tempo total do processo foi gasto com a execução de chamadas ao algoritmo *out-of-kilter*, comportamento esse esperado. Obviamente a qualidade do equipamento utilizado interferiu no número de soluções investigadas. As tabelas 3.2a. e 3.2b. mostram que nenhum dos casos foi resolvido na otimalidade geral (pelo esgotamento da árvore B&B apenas com o LS como poda). Isto aconteceu pela ausência de um bom limite inferior. Uma sugestão seria conhecer este limite via a primeira fase do método de Nobert & Picard (1998), e com um bom limite inferior, se descarta todos aqueles passos com geração de pseudo-ramos que são superiores a este, reduzindo ainda mais a árvore.

Percebemos também que o fato de uma instância ser maior em número de vértices e elos não significa propriamente um aumento na "dificuldade" de resolvê-la. Entendemos aqui por "dificuldade", como o processamento necessário inerente à resolução da instância que se atrela basicamente a sua estrutura topológica. Isto também acontece no problema do caixeiro viajante, onde instâncias com maior número de vértices não requerem necessariamente mais esforço que instâncias com poucos vértices, mas com especial topologia que prejudica o desempenho dos métodos de resolução elaborados. Na tabela 5.2 deixamos indicado em cinza o alto esforço empreendido para se encontrar soluções melhores (LS) na etapa B&B.

As redes selecionadas se encontram em aplicações reais tais como: Coleta de Lixo, Percurso do Fumacê, e outras. Apesar de a literatura ter se ocupado com instâncias em alguns casos bem maiores que as aqui apresentadas, estas instâncias não estão disponíveis para que se avalie o seu grau de dificuldade, ou mesmo se compare de algum modo.

O problema dos tempos conforme o aumento da quantidade de arcos foi traduzido de forma não esperada nas instâncias genéricas, que mostraram o efeito da inclusão de arcos na terminação das soluções do método combinado exato, figuras 15 a 17. Uma oscilação ocorreu nos grafos com 200, 400 e 500 vértices, na qual se atribui ao comportamento desestruturado das instâncias, não permitindo que se generalize o comportamento do método com o crescimento da quantidade de arcos.

O fato de não conseguir provar a otimalidade para algumas instâncias era esperado devido ao custo inerente ao tamanho da árvore *B&B* resultante da parte exata do método e também do custo alto (em tempo) do algoritmo *out-of-kilter* que é executado a cada passagem por um ramo dessa árvore. Porém, mesmo com esse tempo, o método tem a vantagem de poder ser paralisado a qualquer instante e com medidas claras da distância da relaxação linear (LC). Ao final o percurso é de boa qualidade, pois será ótimo em relação a solução incumbente.

No geral o método proposto apresentou um bom desempenho e obteve solução ótima em instâncias mistas que não apresentam a grande maioria de ligações não orientadas. Vale ressaltar que a disponibilidade de memória não é um problema para esse método, instâncias muito grandes (grande número de vértices, elos e arcos) podem ser resolvidas com ele, se desconsiderarmos a exigência para se provar a otimalidade da solução como fator crucial.

6. Aspectos do Sistema Xnês

O **Sistema XNÊS** foi concebido com o intuito de servir como uma ferramenta de exploração do PCC. O ambiente é gráfico, e possui como principal componente um gerador de instâncias de grafos, as quais podem ser descritas conforme as necessidades do usuário. O paradigma de concepção de grafos está baseado no modelo adotado no SisGRAFO (Negreiros, 1996), ou seja, os grafos são definidos através de uma estrutura de dados *hashing* para os vértices, onde para cada célula de vértice está associada uma lista encadeada que define as ligações àquele vértice. Para cada estrutura há uma ligação com vetores e estruturas dinâmicas de atributos (valores) e *bindings* (que são formas, cores e rótulos associados ao elemento vértice ou ligação).

O conceito do sistema é baseado em Modelagem por Grafos, a qual foi proposta inicialmente por Jones (Jones, 1990). Ambientes de Modelagem por Grafos são considerados ambientes de Modelagem Visual Interativa (MVI) (Hurrion, 1986) uma vez que permitem a manipulação direta das instâncias de problema, alterando de forma transparente ao usuário atributos do problema em análise. Neste contexto objetiva-se a obtenção de soluções que possam responder de forma iterativa às necessidades do usuário. Referências de *softwares* que usam modelagem por grafos podem ser encontradas na literatura, como por exemplo: AGILE (Delgrande, 1980), CABRI (Dao *et al.*, 1986), NETPAD (Dean *et al.*, 1992), Stanford Graphbase (Knuth, 1993), Networks (Jones, 1995), NETWK (Donohue, 1995), SisGRAFO (Negreiros, 1996), GIDEN (Owen *et al*., 1999) e GOBLIN (GOBLIN, 2009).

O ambiente geral pode ser visto na figura 18, onde percebemos um conjunto de tarefas no menu, as quais processam arquivos, impressos variados, e definem operações sobre os grafos criados.

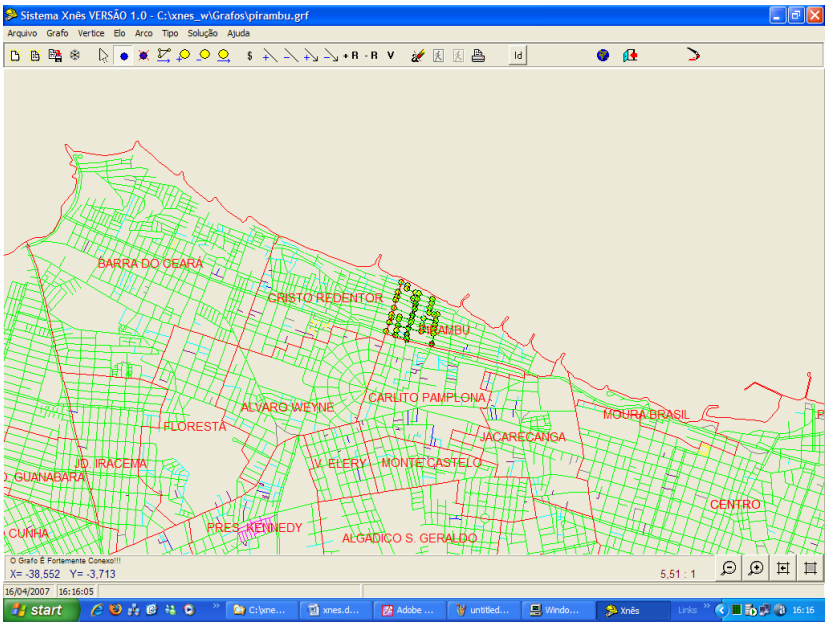

Figura 18 – Ambiente do **Sistema XNÊS** com uma parte da malha viária de Fortaleza/CE.

Entre as tarefas básicas sobre grafos temos: Incluir/Excluir vértice, arco e elo, mover vértice e grafo, e selecionar origem. Além disto, o sistema também possui uma opção de inserir custos nos elos/arcos, de forma manual ou respeitando uma escala na opção custo terrestre, de modo a preservar alguma relação de escala existente entre o mapa de fundo e o grafo gerado. Estas ferramentas permitem que os grafos sejam alterados dinamicamente e visualizados os efeitos das mudanças provocadas em função de variações que possam existir na instância do modelo proposto.

Dentre as opções de geração de soluções (Soluções), o **Sistema XNÊS** checa automaticamente a conectividade do grafo em uso, retornando uma mensagem de erro caso não seja *f*-conexo. Não são listados ou mostrados ao usuário os vértices que estão isolados, porém é indicado um vértice que está desconectado da componente do vértice 1. O sistema também permite a execução dos algoritmos, conforme o modelo em uso (simétrico, direcionado ou misto), processando os algoritmos discutidos neste texto. Após o sucesso do processamento, uma caixa contendo o custo total da rota, limites inferior e superior, e o tempo de processamento é apresentada ao usuário, onde daí por diante, ele pode visualizar o percurso gerado em uma animação dos movimentos do carteiro sobre o grafo em uso. Na animação, é possível que o usuário pare, aumente ou diminua a velocidade de desenvolvimento do percurso através de botões de controle acessíveis na barra de ferramentas de botões rápidos, figura 19.

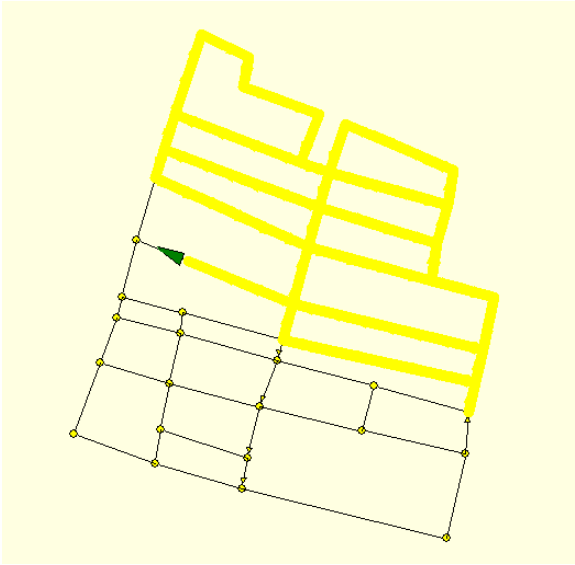

Figura 19 – Uma visão animada do percurso do Carteiro – PCC Misto.

Solucao.Txt

Grafo: C:\XNES_W\Grafos\Pirambu.grf

Tipo: Misto

Nº de Vértices: 40
Nº de Elos: 53
Nº de Arcos: 7

Origem do Percurso: 30

*** PERCURSO ***

30 -> 31 -> 16 -> 12 -> 11 -> 7 -> 8 -> 9 -> 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 5 -> 9 -> 8 -> 10 -> 11 -> 7 -> 35 ->
36 -> 6 -> 36 -> 37 -> 38 -> 6 -> 7 -> 11 -> 34 -> 35 -> 34 -> 33 -> 32 -> 31 -> 32 -> 12 -> 13 -> 10 -> 13 -> 14 ->
15 -> 14 -> 19 -> 18 -> 15 -> 16 -> 17 -> 18 -> 20 -> 21 -> 19 -> 21 -> 27 -> 26 -> 24 -> 20 -> 22 -> 23 -> 25 -> 26 ->
24 -> 23 -> 25 -> 28 -> 29 -> 30 -> 40 -> 17 -> 22 -> 39 -> 40 -> 39 -> 29 -> 30

Custo Total do Percurso: 5920

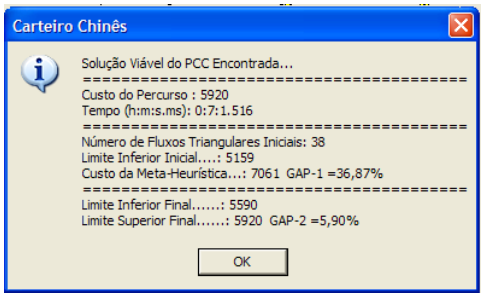

Figura 20 – Tela de solução numérica e relatório do percurso gerado para o grafo misto Pirambu.grf.

Após a animação, o usuário poderá acompanhar o percurso do carteiro em um relatório específico, figura 20, onde é possível identificar a rota euleriana com mais precisão. Além disto, também é impresso um multigrafo que mostra aquelas ligações (e orientações) que devem ser repetidas no percurso, e facilita a visualização das muitas possibilidades de percurso (a partir de uma única solução do CPP), figura 21.

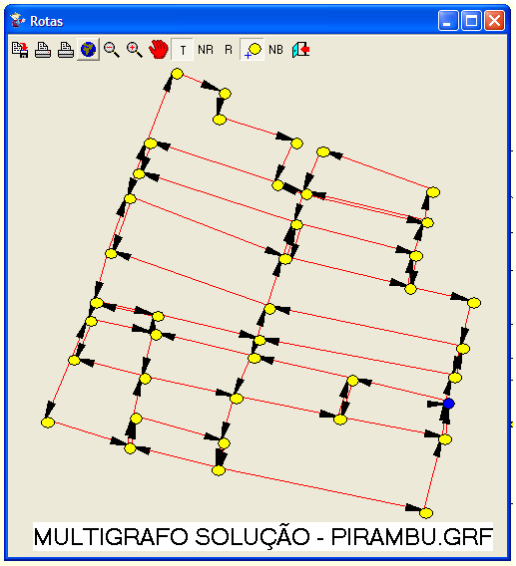

Figura 21 – Multigrafo euleriano criado pelo **Sistema XNÊS**, para o grafo Pirambu.grf.

O **Sistema XNÊS** foi desenvolvido dentro do paradigma da Programação Orientada a Objetos, em ambiente **DELPHI 7.0** (**BORLAND INT.**)-ambiente visual, e em **Borland C++ 4.0** (**BORLAND INT.**), e **MS VISUAL C++** os algoritmos. Roda em ambiente Windows-XP (**MICROSOFT**). Está disponível com o primeiro autor deste trabalho uma versão completa (versão 2.0) do sistema ou em www.lcc.uece.br\~negreiro. Estamos ainda aprimorando este sistema, onde acreditamos que nas próximas versões pretendemos ainda introduzir versões do Carteiro Íngreme (*Windy Postman*), e do Carteiro Rural (*Rural Postman*). Nesta versão, disponibilizamos a possibilidade de inclusão de rótulos nos arcos (nome de ruas, p.ex.), a possibilidade de visualização com *zoom,* barras de rolagem, leitura de arquivos *.dxf*, dentre outras.

Nossa intenção com este sistema foi de dar ao especialista ou mesmo a um aluno de cursos introdutório e avançado de otimização em grafos, o conhecimento deste interessante problema através de uma visualização agradável de suas instâncias e conceitos. Desta forma, é possível sair do caso simétrico para o misto e conhecer as diferenças no esforço da solução, permitir a verificação destes grafos em tempo de execução, ou seja, com a mudança da topologia das instâncias (que é feita de forma automática). É possível entender o que acontece às soluções sem que para isso tenha-se que atualizar os dados da instância base.

7. Comentários Finais

Apresentamos neste trabalho os principais algoritmos do PCC (simétrico, orientado e misto) assim como um sistema computacional (**Sistema XNÊS**), capaz de explorar de forma bastante interativa a resolução deste problema, largamente explorado pela literatura. Tomamos o cuidado de apresentar os procedimentos e formalizá-los, de modo que seja possível compreender melhor o uso do *software* aqui mostrado.

No que concerne aos métodos estudados, fato notório interessante é que em todas as instâncias avaliadas do caso simétrico, o custo da solução ótima encontrado pelo algoritmo polinomial de 1-*Matching* implementado de Burkard & Derigs (1980), realizou sua tarefa de forma imperceptível (menos de 1 segundo) para qualquer das instâncias, invariavelmente com o número de nós e elos das redes estudadas.

Apresentamos também um algoritmo Combinado e Híbrido, que inicia com uma solução meta-heurística e em seguida parte para uma solução numa árvore B&B, para o PCC-Misto assim como as principais formulações deste problema hoje em evidência. Onde a fase híbrida acontece com a introdução de um passo de melhoria a cada nova solução viável encontrada pelo método B&B.

O método combinado GRASP+B&B, usa ideias baseadas em uma transformação do grafo misto em um grafo orientado com pseudo-ramos proposta originalmente por Sherafat. A composição dessas metodologias mostrou ser de grande utilidade. Isto se verificou, pois além de se obter rapidamente um limite para o problema, obteve-se uma metodologia que independe de memória de máquina (tanto quanto possível), e que para qualquer tempo de terminação prematura, tem-se pelo menos os limites (GAP) entre a solução corrente e um limite inferior da instância. Isto dá uma boa ideia de seu grau de otimalidade da solução para o tomador de decisão, provando ser uma forma mais interessante de se apresentar resultados para estes tipos de problemas (NP-Difícil).

Foram explorados exemplos de naturezas diversas, para algumas das áreas sempre sugeridas de aplicação deste problema na literatura (Coleta de lixo, percurso do fumacê, etc.). Em todos os exemplos (instâncias), o desempenho dos métodos exatos aqui estudados foi expressivo aos antes reportados na literatura. Nota-se também que o fato da instância ter um número expressivo de nós e arcos (consideradas de grande porte) não a torna difícil de ser resolvida. Outras características existem e influenciam enormemente no processamento destas instâncias, tais como: a relação entre elos e arcos da rede, a sua topologia em termos de unicursalidade, e a diversidade de valores relacionada ao custo das ligações.

Por fim, vimos alguns dos aspectos gerais de nosso sistema, e as principais visualizações que foram desenvolvidas e as suas vantagens operacionais. O sistema mostra-se robusto, porém se pode ainda incluir novos conceitos em sua interface e algoritmos como o Problema do Carteiro Íngreme (*Windy Postman Problem*) e do Carteiro Rural (*Rural Postman Problem*), e suas derivações.

Trata-se aqui de um trabalho de consolidação de um processo de desenvolvimento e formação teórica promovido por um projeto multi-institucional.

No momento em que Euler completaria 300 anos de existência, e a comunidade científica mundial festeja os seus feitos e contribuições para a humanidade, era de se esperar que esta justa homenagem também se configurasse através de uma equipe de pesquisadores brasileiros envolvidos com estas ideias tão antigas, e seminais da Teoria dos Grafos. A equipe formada por alunos da UFES, liderada pelo professor Francisco Negreiros, e após sua morte conduzida brilhantemente pelos professores Arlindo Gomes Alvarenga e Anilton Salles Garcia, assim como as equipes da UECE e GRAPHVS, liderada pelo professor Marcos Negreiros, sentem-se orgulhosas de participar deste importante momento com esta singela contribuição.

Agradecimentos

Este trabalho vem sendo feito ao longo de dez anos, por uma equipe multi-institucional, Universidade Federal do Espírito Santo (UFES) e Universidade Estadual do Ceará (UECE), assim como pela empresa GRAPHVS Ltda.

Toda a produção inicial deste sistema é resultado de pesquisas de iniciação científica (4), projeto final de graduação (5), dissertação de mestrado (1) e trabalho voluntário de alunos (4) e professores da UFES e UECE. O financiamento de algumas bolsas obtidas ao longo deste trabalho foi feito pelo CNPq (Recém-Doutor. Projeto SisGRAFO), CNPq pelo projeto PQ 202457, CAPES/COFECUB através do projeto 0467/04, e programa ALFA-EUROPEAID através do projeto ALFA II-0457-FA-FCD-F1-FC, FUNCAP/CE, e pela GRAPHVS. Agradecemos a todos os que colaboraram para que o sonho de uma pesquisa multi-institucional de qualidade e participativa do professor Francisco José Negreiros Gomes, fosse enfim concretizado através deste trabalho.

Ao CIRRELT/Universitè de Montreal, através dos professores Gilbert Laporte e Jean-François Cordeau, pela possibilidade de integralizar este documento. Por fim, aos revisores anônimos por terem sugerido boas ideias que melhoraram substancialmente este texto.

Ao professor ***Francisco Negreiros***, nossa saudade e homenagem sincera.

Referências Bibliográficas

(1) Assad, A.A. (2007). Leonhard Euler: A Brief Appreciation. *Networks*, **49**(3), 190-198.

(2) Ahuja, R.K.; Magnanti, T.L. & Orlin, J.B. (1993). *Network Flows, Theory, Algorithms, and Applications*. Prentice-Hall, ed.

(3) Bazaraa, N.S.; Jarvis, J.J. & Sherali, H.D. (1990). *Linear Programming and Network Flow*. New York, John Wiley & Sons, Singapore, 2nd. ed.

(4) Bell, P. (1985). Visual Interactive modeling as an OR technique. *Interfaces*, **15**(4), 26-33.

(5) Bodin, L.D. & Beltrami, E.J. (1974). Networks and Vehicle Routing for Municipal Waste Collection. *Networks*, **4**, 65-94.

(6) Bodin, L. & Kursh, S. (1978). A Computer Assisted System for the Routing and Scheduling of Street Sweepers. *Operations Research*, **26**, 525-537.

(7) Burkard, R.E. & Derigs, U. (1980). Assignment and Matching Problems: Solution Methods with Fortran programs. **In**: *Lecture Notes in Economics and Mathematical Systems*, vol. 184, Spring-Verlag, Berlin.

(8) Christofides, N. (1973). The optimal traversal of a graph. *Omega*, **1**, 719-732.

(9) Christofides, N.; Benavent, E.; Campus, V.; Corberán, A. & Mota, E. (1984). An optimal method for the mixed postman problem. **In**: *System Modelling and Optimization* [edited by P. Thoft-Christensen], Lecture Notes in Control and Information Sciences, vol. 59, Springer, Berlin.

(10) Corberán, A.; Romero, A. & Sanchis, J.M. (1999). Polyhedral results to arc routing problems on mixed graphs. Technical Report, Universitat de Valencia (Spain).

(11) Corberán, A. & Sanchis, J.M. (1994). A polyhedral approach for the rural postman problem. *EJOR*, **79**, 95-114.

(12) Corberán, A. & Sanchis, J.M. (1998). A general Routing Problem Polyhedron: Facets from the RPP and GTSP Polyhedra. *EJOR*, **108**, 538-550.

(13) Corberán, A.; Martí, J.M. & Sanchis, J.M. (2002). A GRASP heuristic for the mixed Chinese Postman problem. *European Journal of Operational Research*, **142**, 70-80.

(14) Dao, M.; Habib, M.; Richard, J.P. & Tallot, D. (1986). CABRI: An interactive system for graph manipulation. **In**: *Graph theoretic concepts in computer science* [edited by Tinhofer & Schimidth], Springer-Verlag, Berlin.

(15) Dean, N.; Movenkamp, M. & Monma, C.L. (1980). NETPAD, An Interactive Graphics System for Network Modeling and Optimization. **In**: *Computer System and Operations Research: New Developments and Their Interfaces* [edited by Balci, Sharda & Zenios], 231-243, Paragon Press, Oxford, UK.

(16) Delgrande, J.P. (1980). A graph-theoretic language extension for an interactive computer graphics environment. *Computer Graphics*, **8**, 13-22.

(17) Derigs, U. (1988). Programming in Networks and Graphs – On the combinatorial background and near-equivalence of network flow and matching algorithms. **In**: *Lecture Notes in Economics and Mathematical Systems* [edited by M. Beckmann and W. Krelle,], vol. 300, Springer-Verlag.

(18) Donohue, J. (1995). NETWK, Problem-solving tool for network and enhanced network models. *OR/MS Today*, december, 46-49.

(19) Edmonds, J. & Johnson, E. (1973). Matching, Euler tours and the Chinese postman problem. *Mathematical Programming*, **5**, 88-124.

(20) Eiselt, H.A.; Gendreau, M. & Laporte, G. (1995). Arc Routing Problems, Part I: The Chinese Postman Problem. *Operations Research*, **43**(2), 231-242.

(21) Eiselt, H.A.; Gendreau, M. & Laporte, G. (1995). Arc-routing problems, Part 2: The Rural Postman Problem. *Operations Research*, **43**, 399-414.

(22) Euler, L. (1736). Solutio problematics ad geometrian situs pertinentis. *Comentarii Academae Scientarum, Petropolinae*, **8**, 128-140.

(23) Evans, J.R. & Minieka, E. (1978). *Optimization Algorithms for Networks and Graphs.* 2nd edition, Marcel Dekker, Inc.

(24) Feo, T. & Resende, M. (1995). Greedy randomized adaptative search procedures. *Journal of Global Optimization Algorithms*, **6**, 109-133.

(25) Ford, L.R. & Fulkerson, D.R. (1962). *Flows in Networks*. Princeton University Press, Princeton NJ (USA).

(26) Gabow, H.N. (1975). An Efficient Implementation of Edmonds's maximum matching algorithm. *Journal of the ACM*, **23**, 221-234.

(27) GOBLIN (2009). GOBLIN: A Graph Object Library for Network Programming Problems. Disponível em <http://goblin2.sourceforge.net/> (GNU LPL). Acessos em 05/2003, 03/2009.

(28) Grotschel, M. & Win, Z. (1992). A Cutting-Plane Algorithm for the Windy Postman Problem. *Mathematical Programming*, **55**, 339-358.

(29) Groves, G.W. & van Vuuren, J.H. (2005). Efficient heuristics for the rural postman problem. *ORION*, **21**(1), 33-51.

(30) Gribkovskaia, I.; Halskau, Ø. & Laporte, G. (2007). The Bridges of Königsberg – A Historical Perspective. *Networks*, **49**(3), 199-203.

(31) Heske, A. (1978). Die Bestimmung maximaler Matchings in beliebigen Graphen. Diploma Thesis, Mathematisches Institut, Universität zu Köln.

(32) Heierholzer, C. (1873). Uber die moglichkeit, einen linienzung ohne wiederholing und ohne unterbrechung zu umfahren. *Mathematische Annalen*, **IV**, 30-32.

(33) Hitchcock, F.L. (1941). The distribution of a product from several sources to numerous facilities. *Journal of Mathematical Physics*, **20**, 224-230.

(34) Hurrion, R.D. (1986). Visual Interactive Modeling. *EJOR*, **23**, 281-287.

(35) Jones, C.V. (1990). An introduction to Graph-Based Modeling Systems, Part I: Overview. *ORSA Journal on Computing*, **2**, 136-1151.

(36) Jones, C.V. (1996). *Visualization and Optimization*. Kluwer Academics Publishes.

(37) Knuth, D.E. (1993). *The Stanford Graphbase, A Platform for Combinatorial Computing*. Stanford University, Addison-Wesley.

(38) Koopmans, T.C. (1947). Optimum utilization of the transportation system. *Econometrica*, **17**, 3-4.

(39) Kwan, Mei-Ko (1962). Graphic programming using odd and even points. *Chinese Mathematics*, **1**, 273-277.

(40) Lenstra, J. & Rinooy Kan, A. (1981). Complexity of Vehicle Routing and Scheduling Problems. *Networks*, **11**, 221-227.

(41) Martinez, F.J.Z. (2003). The Postman Problem on Mixed Graphs. PhD thesis presented to the University of Waterloo, Waterloo, Ontario, Canada, 2003.

(42) Minieka, E. (1979). The Chinese Postman Problem for Mixed Networks. *Management Science*, **23**(7), 643-658.

(43) Negreiros Gomes, M.J. (1990). O Problema de Planejamento e Percurso de Veículos na Coleta de Lixo Urbano Domiciliar. MSc. Dissertation UFRJ – COPPE/Sistemas.

(44) Negreiros Gomes, M.J. (1996). Contribuições para otimização em grafos e problemas de percursos de veículos: Sistema SisGRAFO. DSc. Thesis UFRJ – COPPE/Sistemas.

(45) Negreiros Gomes, M.J. & Coelho Jr., W.C. (2007). A hybrid metaheuristic for the (mixed) windy postman problem. Technical Report, LCC-UECE.

(46) Nobert, Y. & Picard, J-C. (1996). An optimal algorithm for the Mixed Chinese Postman Problem. *Networks*, **27**, 95-108.

(47) Orlin, J.B. (1988). A faster strongly polynomial minimum cost flow algorithm. *Proceedings of the 20th ACM Symposium on the Theory of Computing*, 377-387.

(48) Owen, J.H.; Coullard, C.R. & Dilworth, D.S. (1999). GIDEN, A graphical environment for network optimization. Disponível em <http://giden.nwu.edu/>.

(49) Pearn, W.L. & Chou, J.B. (1999). Improved solutions for the mixed chinese postman problem on mixed networks. *Computers & Operations Research*, **26**, 819-827.

(50) Plotkin, S. & Tardos, É. (1990). Improved dual network simplex. *Proceedings of the First ACM-SIAM Symposium on Discrete Algorithms*, 367-376.

(51) Sherafat, H. (1988). Uma Solução para o Problema do Carteiro Chinês Misto. *Anais do IV CLAIO – XXI SBPO*, 157-170, Rio de Janeiro.

(52) TRANSCAD (2007). <http://www.logitconsultoria.com.br/pd_transcad.htm>. Site consultado em 15/04/2007.

(53) Wolsey, L.A. (1998). *Integer Programming*. Wiley-Interscience Series in Discrete Mathematics and Optimization.

(54) Yaxiong, L. & Yongchang, Z. (1988). A new algorithm for the direct chinese postman problem. *Computers & Operations Research*, **15**, 577-584.