

UM TRATAMENTO GRASP HÍBRIDO DO PROBLEMA DO CARTEIRO CHINÊS MISTO (PCCM) DESTINADO A OTIMIZAÇÃO DE ROTAS PARA COLETA DO LIXO DOMÉSTICO

Frederico G. Paes, Gudelia M. de Arica

Universidade Estadual do Norte Fluminense/CCT
Laboratório de Engenharia de Produção
Av. Alberto Lamego, 2000, 28013-600, Campos dos Goytacazes-RJ
E-mail: fredgal@uenf.br, gudelia@uenf.br

O objetivo dos problemas de roteamento em arcos (PRA's) é determinar um circuito de custo mínimo, também chamado de rota, em um grafo com ou sem restrições, tal que todos os arcos sejam atravessados ao menos uma vez.

Estes problemas surgem em diversos contextos práticos do setor público e privado, onde há a necessidade de otimizar a rota como, por exemplo, na distribuição de cartas, na coleta do lixo doméstico, na varrição e lavagem de ruas, na inspeção de redes elétricas, de água ou gás, entre outros.

Dentre os principais problemas de roteamento em arcos, pode-se citar o problema do carteiro chinês (PCC) que, de acordo com a natureza da rede, classifica-se em: o problema do carteiro chinês não direcionado (PCCND), o problema do carteiro chinês direcionado (PCCD) e o problema do carteiro chinês misto (PCCM). Portanto, este artigo aborda uma proposta computacional para o PCCM destinado a otimizar a rota executada por um veículo em um determinado setor de coleta.

1- Notações e Definição do PCCM

Dado um vértice $i \in V$, d_i^- (grau de entrada) representará o número de arcos que entram em i , d_i^+ (grau de saída) o número de arcos que deixam i e d_i (grau) o número de arcos e arestas incidentes em i . Um grafo misto $G = (V, E \cup A)$ é dito par se todos seus vértices tem grau par, é dito simétrico se $d_i^- = d_i^+$ para cada vértice $i \in V$ e é dito ser balanceado se, para qualquer subconjunto S de vértices, a diferença entre o número de arcos direcionados de S para $V \setminus S$ e o número de arcos direcionados de $V \setminus S$ para S não é maior que o número de arestas (não direcionadas) unindo S e $V \setminus S$ [3][1].

Uma condição que deve ser ressaltada é a de grafo *euleriano*. Um grafo conectado é *euleriano*, se existe um caminho fechado em G contendo cada arco exatamente uma vez e cada vértice ao menos uma

vez. Ford e Fulkerson [3] estabeleceram as condições necessárias e suficientes para que um grafo conectado seja *euleriano*:

1. Seja G não direcionado, G é *euleriano* se e somente se todo vértice do grafo tem grau par, isto é, um número par de arestas incidentes;
2. Seja G direcionado, G é *euleriano* se e somente se o número de arcos que entram e saem de cada vértice são iguais. Em outras palavras, o grafo deve ser simétrico;
3. Seja G misto, G é *euleriano* se e somente se todo vértice tem grau par e, além disso, é balanceado.

Note que, se um grafo misto G é par e simétrico então G também é balanceado e, portanto, *euleriano*. Portanto, considerando um grafo fortemente conectado G , com custos não negativos associados aos seus arcos ou arestas, o problema do carteiro chinês (PCC) consiste em encontrar uma rota de custo mínimo (caminho fechado) atravessando, ao menos uma vez, todo arco ou aresta de G . Este problema possui complexidade polinomial para grafos não direcionados (todas suas ligações são arestas) e grafos direcionados (todas suas ligações são arcos) [2]. Contudo, se o PCC for definido sobre um grafo misto torna-se NP-Completo [1]. A definição formal do problema é:

Dado um grafo misto fortemente conectado $G = (V, E \cup A)$, sendo $V = \{v_1, v_2, \dots, v_n\}$ o conjunto de nós ou vértices, E o conjunto de arestas, $A = \{(v_i, v_j) : v_i, v_j \in V \text{ e } i \neq j\}$ o conjunto de arcos e um custo não-negativo c_e (que pode ser a distância ou tamanho do arco/aresta) para cada $e \in E \cup A$, o PCCM consiste em encontrar uma rota de custo mínimo passando através de cada ligação $e \in E \cup A$ ao menos uma vez.

Se um grafo G (direcionado, não direcionado ou misto) é *euleriano* existe uma rota passando por cada

ligação de G exatamente uma vez. Obviamente esta rota será ótima e poderá ser facilmente encontrada. Neste caso, o próprio G pode ser considerado como a solução para o **PCC**. Por outro lado, se um grafo (direcionado, não direcionado ou misto) não é *euleriano*, o **PCC** pode ser formulado como um problema, cujo objetivo é encontrar um conjunto de cópias de ligações com o menor custo, tal que, quando adicionadas a G um grafo *euleriano* seja obtido. Assim, o grafo aumentado formado por G mais as cópias das ligações adicionadas, pode ser considerado como a solução do problema. Na Figura 1, a cópia de menor custo adicionada está representada pela aresta tracejada.

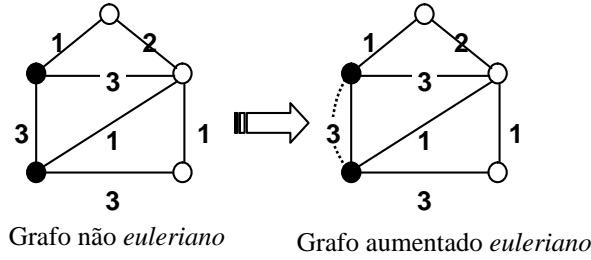


Figura 1: Transformação de um grafo não direcionado.

Portanto, para resolver um **PCCM** deve-se encontrar os arcos e arestas de custo mínimo que irão aumentar G de maneira conveniente, transformando-o num grafo *euleriano*, para então determinar um circuito no grafo aumentado. Se por outro lado, já se sabe que G é um grafo *euleriano*, o problema reduz-se a determinar um circuito em G . Isso pode ser obtido em três fases: 1- designar direções para algumas arestas tornando G simétrico; 2- designar direções para as arestas restantes; 3 - determinar um circuito em G . Para gerar um grafo simétrico de G , pode-se aplicar o procedimento proposto por Ford e Fulkerson [3]. Uma vez obtido um grafo completamente orientado, utiliza-se um algoritmo apropriado para determinar um circuito *euleriano* em G .

O **PCCM** então, fica reduzido a determinar o aumento de arcos e arestas com menor custo no grafo original G , de forma a tornar o grafo *euleriano*. Diversos autores têm utilizado Programação Linear Inteira (**PLI**) para determinar este aumento de menor custo em grafos mistos de forma exata. Dentre estes, pode-se citar Grötschel e Win [5], Kappauf e Koehler [6] que apresentaram formulações praticamente idênticas, diferindo apenas no nome das variáveis. Uma formulação em especial, devido a Ralphs [12], será destacada por ter sido utilizada na obtenção dos valores ótimos dos grafos testados. Para entender o modelo formulado por Ralphs, algumas notações

devem ser esclarecidas. Em um grafo misto $G=(V,A\cup E)$, \hat{E} e \check{E} representam conjuntos distintos de arestas que serão orientadas, um contendo cópias das arestas pertencente a E em uma dada direção e o outro, contendo cópias das mesmas arestas, orientadas em direção oposta. Então, se e é uma cópia de uma aresta orientada pertencente a \hat{E} ou \check{E} , então \tilde{e} é a outra cópia correspondente. A formulação de **PLI** para o **PCCM** é a seguinte:

2- Formulação para o PCCM

$$\text{Minimize } \sum_{s \in A \cup \check{E} \cup \hat{E}} c_s x_s \quad (1)$$

s.t.

$$y'_e + y'_e \geq 1 \quad \forall e \in E \quad (2)$$

$$x_s = y'_s + y_s \quad \forall s \in A \cup \check{E} \cup \hat{E} \quad (3)$$

$$\sum_{s \in \delta^+(v)} x_s - \sum_{s \in \delta^-(v)} x_s = 0 \quad \forall v \in V \quad (4)$$

$$y'_a = 1 \quad \forall a \in A \quad (5)$$

$$y'_e \in \{0,1\} \quad \forall e \in \check{E} \cup \hat{E} \quad (6)$$

$$y_s \geq 0, \text{ inteiro} \quad \forall s \in A \cup \check{E} \cup \hat{E}. \quad (7)$$

Modelo 1: Formulação de PLI.

Na formulação acima, pode-se pensar em cada solução viável como um grafo aumentado particular. Para cada arco, y'_a representa a ‘cópia original’ do arco contido em A e y_a representa o número de ‘cópias adicionais’ incluídas no grafo aumentado. Desta forma, as restrições (5) representam a necessidade de incluir ao menos uma cópia de cada arco no grafo aumentado. As restrições (3) nos fornecem o total de arcos unindo quaisquer dois vértices distintos no grafo aumentado. Já foi visto anteriormente o significado de \hat{E} e \check{E} . Portanto as variáveis y'_e e y'_e representam as cópias orientadas de uma aresta tomadas em cada direção, enquanto y_e e y_e representam cópias orientadas adicionais da mesma aresta incluída no grafo aumentado.

As restrições (2) representam a necessidade de que haja ao menos uma cópia de cada aresta no grafo aumentado. As restrições (4) garantem que o grau de cada vértice do grafo aumentado seja zero, uma vez que $\delta^-(v)$ (conjunto de arcos e arestas orientadas que entram em v) e $\delta^+(v)$ (conjunto de arcos e arestas orientadas que saem de v) referem-se respectivamente

ao grau de entrada e saída dos nós no grafo aumentado.

Nobert e Picard [8] desenvolveram um algoritmo de plano de corte baseado em **PL** para resolver o **PCCM** de forma exata. Contudo, todos estes métodos são computacionalmente ineficientes, pois apenas problemas de médio e pequeno porte podem ser resolvidos otimamente [11].

Uma heurística para o **PCCM** foi sugerida por Edmonds e Johnson [2] para resolver o **PCCM** aproximadamente. Essa heurística consiste basicamente de duas fases. A fase 1 converte o grafo original em um grafo par, por considerar cada arco como uma aresta, aplicando o algoritmo de Emparelhamento de Custo Mínimo. A fase 2 transforma o grafo obtido da fase 1 em um grafo simétrico por meio do algoritmo de Fluxo de custo Mínimo. Desde que o grafo simétrico transformado não é necessariamente mantido par, a rota pode não ser construída. Frederickson [10] modificou a heurística adicionando uma nova fase (fase 3). A nova fase transforma novamente o grafo resultante em par, podendo, assim, ser construída melhor rota. Esta heurística foi chamado de MIXED 1 [10]. Esta heurística gera boas soluções satisfazendo as condições necessárias e suficientes para um grafo ser *euleriano*.

Frederickson mostrou ainda que a complexidade computacional do algoritmo MIXED 1 é $O(\max\{|V|^3, |A|(\max\{|A|, |E|\})^2\})$, onde $|V|$ = número de nós, $|A|$ = número de arcos e $|E|$ = número de arestas do grafo. Um outro algoritmo, que praticamente faz o inverso do algoritmo MIXED 1, é o algoritmo MIXED 2. Este, primeiro obtém um grafo simétrico e posteriormente o transforma em par. A complexidade computacional do MIXED 2 é a mesma do MIXED 1.

3- O Algoritmo Proposto

O algoritmo utilizado para resolver o **PCCM** e otimizar as rotas dos veículos foi um algoritmo híbrido baseado na metaheurística **GRASP**, [9]. As fases componentes deste algoritmo: fase de construção e fase de busca local; serão descritas separadamente de maneira que sejam identificadas suas peculiaridades.

3.1- Fase de Construção

O objetivo da fase de construção é orientar as arestas do grafo a fim de obter um grafo direcionado, e com o auxílio de um algoritmo exato que resolva o Problema de Fluxo de Custo Mínimo (**PFCM**), torná-lo simétrico.

Neste caso, o **PFCM** é resolvido em um grafo direcionado com ofertas e demandas computados nos vértices. A rotina que resolve o **PFCM** foi implementada com base no algoritmo Simplex para redes com eficiência polinomial, $O(mn)$ [4].

Uma adaptação incorporada foi a criação de um vetor chamado *Listael*, o qual armazena os arcos pertencente ao ciclo formado pelo arco entrante, que possuem fluxo zero. Uma das características dos grafos com grau igual a zero é a existência de arcos com fluxo $f_{ij} = 0$ na árvore básica. Portanto, no momento da seleção do arco candidato a deixar a base ($\min\{f_{ij}/i, j \in V\}$ dentre os arcos do ciclo), surge mais de um arco com $f_{ij} = 0$. Para resolver isto, todos os arcos pertencentes ao ciclo formado pelo arco entrante, são armazenados no vetor *Listael* com seus respectivos valores de fluxo. Então, o primeiro arco da lista com $f_{ij} = 0$ será selecionado como candidato a deixar a base, opção esta inspirada no método de Bland que evita conseqüências negativas na presença de uma solução viável degenerada [7].

A solução ótima da rotina descrita acima é uma *Árvore*. Estes arcos deverão ser acrescentados ao grafo original de forma a torná-lo simétrico. Esta rotina representa a parte principal do programa, uma vez que, após sua execução, o grafo terá se tornado num grafo *euleriano* ou não, que é o caso da ocorrência de ciclo negativo. Neste caso, o programa é encerrado, indicando que o problema é inviável. Caso contrário obtêm-se uma solução para o **PCCM** no grafo misto original.

A fase de construção inicia criando-se uma lista de arestas não orientadas (U), contendo todas as arestas do grafo ($U = E$). A cada passo da fase de construção, uma aresta é selecionada aleatoriamente e orientada. O conjunto de arestas orientadas será denotado por E_d . A partir de agora, $d(v)$ será a diferença entre o número de arcos e arestas orientadas que entram em v e o número de arcos e arestas orientadas que saem de v , ou seja, $d(v) = d(v)^- - d(v)^+$. Sem causar confusão, $d(v)$ será chamado de grau do vértice v . Portanto, $d(v) > 0$ indicará que existe uma oferta no vértice v , $d(v) < 0$ indicará uma demanda, enquanto $d(v) = 0$ indicará que v é um vértice de passagem.

A função de avaliação gulosa $w(i, j)$ de determinada aresta (i, j) , representa a conveniência de orientá-la, de forma a aproximar o grafo de um grafo simétrico. O valor de $w(i, j)$ para determinada aresta (i, j) será o seu peso. Considere, por exemplo, uma aresta (i, j) tal que $d(i) = 3$ e $d(j) = -4$. A orientação mais apropriada para (i, j) seria de i para j , e não ao contrário (Figura 2). Por outro lado, se $d(i) = 2$ e $d(j) = 2$, não existe um sentido mais apropriado que outro, portanto a orientação será aleatória.

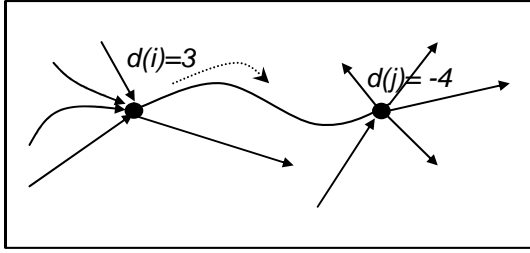


Figura 2: Orientação de uma aresta (i, j) .

Considerando ε um número positivo muito pequeno, pode-se considerar para a função $w(i, j)$, da aresta (i, j) os seguintes casos:

Caso 1: se $d(i).d(j) > 0$ então $w(i, j) = -|d(i) + d(j)|$.

Caso 2: se $d(i).d(j) < 0$ então $w(i, j) = |d(i) - d(j)|$.

Caso 3: se $d(i)=0$ e $d(j)=0$ então $w(i, j) = -\varepsilon$.

Caso 4: se $d(i)=0$ ou $d(j)=0$ então $w(i, j) = \varepsilon$.

Portanto, após todos os pesos $w(i, j)$ terem sido calculados, o conjunto U será ordenado em ordem decrescente de pesos (w). A cada passo da fase de construção, uma aresta (i, j) é aleatoriamente selecionada entre os λ primeiros elementos de U , onde λ é o tamanho da **LCR**. Então a aresta (i, j) é orientada do vértice com maior grau para o vértice com menor grau. Se i e j tiverem o mesmo grau, então (i, j) é aleatoriamente orientado. Os conjuntos U e E_d são atualizados através da retirada de (i, j) de U e a inserção desta em E_d . Do mesmo modo $d(i)$, $d(j)$ e $w(i, j)$ são também atualizados. A fase de construção termina quando todas as arestas tiverem sido selecionadas e orientadas. Conseqüentemente, quanto maior o número de arestas do grafo, mais tempo levará para a fase de construção ser finalizada. Desta forma, a fase de construção termina após $|E|$ passos. Então, o **PFCM** com demandas e ofertas computadas com respeito aos arcos em $A \cup E_d$ é resolvido em um grafo direcionado $G_F = (V, A \cup E_d \cup E_d')$, onde E_d' é o conjunto de arcos paralelos àqueles em E_d com o mesmo custo, porém com direções opostas. Adicionando os arcos pertencentes à solução do **PFCM** ao grafo $G = (V, A \cup E_d)$, obtém-se um digrafo $G_S = (V, A \cup E_d \cup A_F)$, simétrico e *euleriano*. Associado a este grafo está uma rota T com custo $Z(T)$ correspondente a uma solução do grafo misto original. Após estes passos segue-se a fase de melhoria.

3.2- Fase de Melhoria

Nesta fase aplica-se um procedimento de busca local objetivando encontrar um ótimo local que seja melhor que a solução construída. O procedimento

parte então da rota T , obtida na fase de construção sobre o grafo que é a solução do **PCCM**. Este grafo, chamado de G_S , é representado por um digrafo (grafo orientado) fortemente conectado, contendo todos os arcos originais e arestas orientadas do grafo e uma ou mais cópias destas ligações. Deve-se notar que, no grafo solução existirá ou um arco simples, dois arcos com direções opostas ou dois ou mais arcos, todos eles com a mesma direção, associado a cada aresta original $e \in E$.

Dados dois vértices i e j unindo um caminho de cópias, pode-se efetuar a melhoria, através da substituição destas cópias pelas cópias dos arcos do caminho mínimo em G unindo tais vértices. Portanto a melhoria, definida como $m(i, j)$, é a diferença entre o valor da função objetivo antes e após a substituição. Mais ainda, ela pode ser calculada como o custo do caminho deletado menos o custo do caminho mínimo adicionado. Cada passo da fase de melhoria, consiste em selecionar um par de vértices que estejam unidos por um caminho de cópias. Para tanto, deve-se explicitar tais pares construindo um novo grafo misto $G' = (V, E', A')$, derivado do grafo solução G_S , assim:

Para cada arco original $(i, j) \in A$ coloque $n - 1$ cópias dele em A' , onde n é o número de vezes que ele aparece em G_S . Para cada aresta $e \in E$, tendo exatamente dois arcos com direções opostas associados em G_S , coloque uma aresta em E' , caso contrário, $n - 1$ cópias de seus arcos associados são adicionados a A' .

Então, computa-se estas componentes conectadas do grafo G' e reduz-se a busca a pares de vértices dentro de cada componente. No intuito de reduzir o esforço computacional, esta busca estará restrita apenas a vértices adjacentes a somente um outro vértice. Se o valor da melhoria for positivo, a troca é realizada, caso contrário, será rejeitada. A fase de melhoria termina quando todos os pares de vértices forem revisados e nenhuma melhoria possa ser mais realizada.

Após completar um número de iterações maior do que $NITER$ ($Nmáx$), a melhor solução encontrada é retornada como solução final. Observa-se que os parâmetros $NITER$ e λ (tamanho da **LCR**) são os únicos a serem ajustados no algoritmo. O pseudocódigo do algoritmo **GRASP** é apresentado a seguir:

Procedimento *Otimiza_Coleta*($G(V, A \cup E)$, $Nmáx$, λ);

- 1 iter \leftarrow 0;
- 2 Niter \leftarrow $Nmáx$;
- 3 enquanto (iter < Niter) faça
- 4 iter \leftarrow iter + 1;

seja $E_d = \emptyset$ e $U = E$. Compute o $peso(e) = w(i,j)$; para todo $e = (i,j) \in U$;

- 5 enquanto ($U \neq \emptyset$) faça *{Fase de Construção}*
- 6 Considere as λ arestas em U com maior peso. (λ é o tamanho da *L.C.R.*);
- 7 Selecione uma aresta (i,j) aleatoriamente, oriente-a, delete-a de U e adicione a E_d ;
- 8 Atualize os pesos de todas as arestas em U incidentes em i e j ;
- 9 fim-enquanto;
- 10 calcule os graus (oferta/demanda) dos vértices em G_F ;
- 11 encontre o **FCM** em G_F , com os graus calculados em (10);
- 12 adicionando a $A \cup E_d$ uma cópia de cada arco e cada aresta orientada usada pelo fluxo (11), obtém-se um grafo **Euleriano Orientado** com uma rota T de custo $z(T)$;

13 enquanto (houver melhoria) faça *{Fase de Melhoria}*

- 14 remova de T as cópias das arestas que apareça mais que duas vezes em T e tem rótulos de orientação opostos. Atualize $z(T)$;
- 15 construa o Grafo Misto $G' = (V, E', A')$ e compute seus componentes conectados;
- 16 para cada um destes componentes conectados faça:

encontre u e v tal que o tamanho do menor caminho de u até v em G , chamado de ϕ , seja menor que o tamanho do menor caminho de u até v , ϕ , neste componente conectado; Remova de T as cópias dos links em ϕ e adicione a T uma cópia de cada link em ϕ . **Melhoria** \leftarrow tamanho (ϕ) - tamanho(ϕ); $z(T) \leftarrow z(T) - \text{melhoria}$;

17 fim-enquanto;

18 se ($z(T) < z_{\text{best}}$) então *{Teste Final}*

19 $z_{\text{best}} \leftarrow z(T)$;

20 $T_{\text{best}} \leftarrow T$;

21 fim-enquanto;

22 Retorne z_{best} e T_{best} .

Fim *Otimiza_Coleta*.

Algoritmo 1: Pseudocódigo do algoritmo do **PCCM** baseado na **GRASP**.

4- Formulação do Modelo e Testes

A formulação do **PCCM** apresentada no Modelo 1, é uma formulação de **PLI** proposta por Ralphs para resolver o problema de forma exata. Tal modelo foi utilizado em três grafos, com 30%, 50% e 70% do total de ligações compostas por arcos. Estes grafos

estão representados na Tabela 1 por meio de dois números, onde o 1º indica o número de vértices e o segundo o número de ligações (arcos e arestas) do grafo. É importante destacar que cada aresta está sendo considerada como dois arcos. Os testes realizados para a obtenção da solução ótima em tais grafos foram realizados utilizando o software **LINDO®** 6.01 em um PC com processador AMD Duron 1200 Ghz e 248 Mb de memória e observou-se um elevado tempo de processamento para os grafos com maior número de variáveis. Além disso, nos grafos com 30% e 50% de arcos não foi possível alcançar a solução ótima devido a uma falha no sistema acusando estouro de memória.

Através de uma adaptação feita neste modelo, foi possível obter a solução ótima nos grafos testados. Tal adaptação é um novo modelo de **PLI** para o **PCCM**, onde são consideradas algumas modificações. Sabendo-se que uma cópia orientada de cada aresta e de cada arco será incluída no grafo aumentado ótimo, pode-se considerar o custo destas cópias como constantes na função objetivo (8). Fazendo isso, quando o problema for resolvido só será considerado o custo de aumento do grafo, gerando uma redução no número de variáveis, pois serão suprimidas as variáveis y'_a referentes às cópias originais de cada arco $a \in A$. As restrições (2) passam a ser $y'_e + y''_e = 1$, forçando cada aresta ser orientada em uma dada direção. O novo modelo, que é uma formulação de **PLI** com algumas variáveis restritas a valores **0-1**, fica então da seguinte forma:

$$\text{Minimize } \sum_{s \in A \cup \check{E} \cup \hat{E}} c_s y_s + \sum_{s \in A \cup E} c_s \quad (8)$$

s.t.

$$y'_e + y''_e = 1 \quad \forall e \in E, \quad (9)$$

$$x_e = y'_e + y''_e \quad \forall e \in \check{E} \cup \hat{E}, \quad (10)$$

$$x_a = y_a \quad \forall a \in A, \quad (11)$$

$$\sum_{s \in S^+(v)} x_s - \sum_{s \in S^-(v)} x_s = d(v) \quad \forall v \in V, \quad (12)$$

$$y'_e \in \{0, 1\} \quad \forall e \in \check{E} \cup \hat{E}, \quad (13)$$

$$y_s \geq 0, \text{ inteiro} \quad \forall s \in A \cup \check{E} \cup \hat{E}. \quad (14)$$

Modelo 2: Nova formulação de **PLI**.

Nesta formulação, as restrições (10) e (11) podem ser eliminadas do modelo sem nenhum prejuízo para a solução final, uma vez que estas só indicam o número de ligações que unem cada par de nós.

As restrições (12) garantem que as ligações acrescentadas ao grafo atendam as ofertas ou demandas nos vértices, isto é, seus graus sejam anulados tornando-o assim em um grafo simétrico. Recorde que $d(v)$ representa os graus dos vértices.

Em uma das iterações, o novo modelo forneceu a solução ótima em 1 minuto e 54 segundos para o caso do grafo com 50% das ligações formadas por arcos e no caso do grafo com 30%, o ótimo foi obtido em 4 minutos e 3 segundos. No caso do grafo de 70% a solução ótima foi obtida com um tempo menor que 1 segundo.

De posse dos valores ótimos de cada grafo, foi possível efetuar uma comparação com os valores encontrados pelo programa OTIMIZA_COLETA gerado a partir do Algoritmo 1. A Tabela 1 apresenta diversos valores encontrados pelo programa OTIMIZA_COLETA para diferentes tamanhos de λ em uma instância com 44 vértices. Procurou-se tomar valores para λ , tais que caracterizassem um algoritmo totalmente guloso ($\lambda = 1$) e totalmente aleatório ($\lambda = n^\circ$ de arestas do grafo testado), e alguns valores intermediários a estas duas situações. Para efetuar estes testes fixou-se o número de iterações GRASP em 1000, isto é, $N_{máx}=1000$.

Tabela 1: Média das soluções encontradas para as instâncias **44:118**, **44:108** e **44:91**.

| Grafo | $\lambda=1$ | $\lambda=5$ | $\lambda=10$ |
|---------------------------------|-----------------------|-------------------------|------------------|
| | Z(T) Médio | Z(T) Médio | Z(T) Médio |
| 44:118 (30%) Z* = 274 | 301 12 seg. | 287,6 12 seg. | 290,2 12 seg. |
| 44:108 (50%) Z* = 283 | 309,2 11 seg. | 291,4 11 seg. | 293,8 11 seg. |
| 44:91 (70%) Z* = 385 | 407 10 seg. | 398,2 09 seg. | 395,6 09 seg. |
| Grafo | $\lambda=20$ | $\lambda=38$ | $\lambda=48$ |
| | Z(T) Médio | Z(T) Médio | Z(T) Médio |
| 44:118 (30%) Z* = 274 | 283 12 seg. | 281 12 seg. | 287,2 12 seg. |
| 44:108 (50%) Z* = 283 | 298 11 seg. | 291,8 11 seg. | - |
| 44:91 (70%) Z* = 385 | 390 09 seg. | - | - |

Além dos testes comparativos dos grafos apresentados na Tabela 1, foram realizados testes em grafos com 64, 84 e 200 vértices, mantendo a mesma variação do número de arcos, para verificar o desempenho do algoritmo. A média das soluções obtidas para cada grafo e seus respectivos tamanhos da LCR (valores de λ), são apresentadas nos Gráficos 1, 2 e 3.

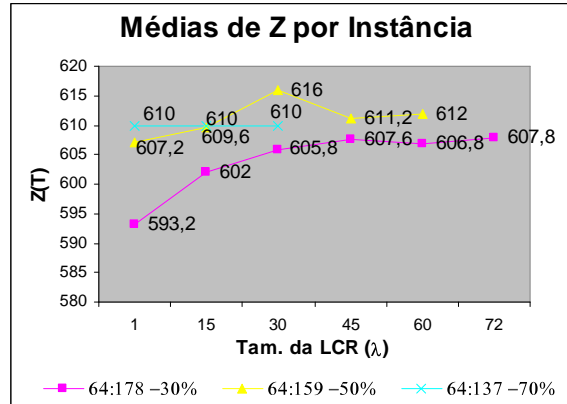


Gráfico 1: Grafos com 64 vértices.

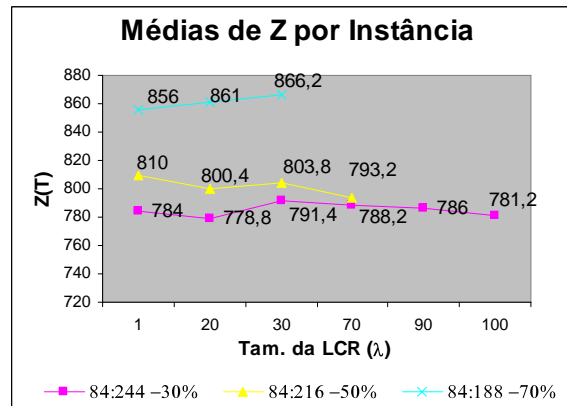


Gráfico 2: Redes com 84 vértices.

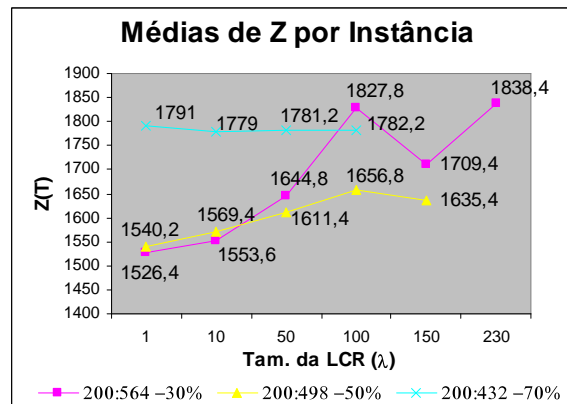


Gráfico 3: Redes com 200 vértices.

5- Conclusão

Os Problemas de Roteamento em Arcos (**PRA's**) possuem diversas aplicações práticas, conforme foi apresentado.

O cálculo de uma solução para o caso do problema do Carteiro Chinês Misto (**PCCM**), explorado neste artigo, é classificado como sendo de complexidade computacional NP-Completo, pois não se conhece um algoritmo que o resolva em tempo eficiente (de ordem polinomial).

Verificou-se durante a etapa de testes, que quando se calcula a solução ótima do **PCCM** por meio de um método exato aplicado ao modelo de **PLI**, o tempo de processamento aumenta, na medida que um setor de coleta apresenta acréscimo no número de ruas de mão-dupla (grafo com maior número de arestas).

Por outro lado, o algoritmo heurístico implementado mostrou-se eficiente quanto ao tempo de resposta, e pelo fato de ter como entrada alguns poucos parâmetros. Além disso, algumas soluções ficaram próximas da solução ótima. Como exemplo, foi encontrada uma solução $Z(T) = 277$ para a instância **44_118** utilizando $\lambda = 5$, bem próxima do ótimo (274) com um tempo de processamento de 12 segundos. Na Tabela 1 a menor média para esta instância foi 281 para um $\lambda = 38$. O modelo exato por sua vez, aplicado à mesma instância encontrou o referido ótimo em 4 minutos e 3 segundos.

Bibliografia

- [1] A. Corberán, R. Martí, J.M. Sanchis, A GRASP heuristic for the mixed Chinese postman problem. *European Journal of Operational Research*, 142 (2002) 70 – 80.
- [2] J. Edmonds, E.L. Johnson, Matching, Euler tours and the Chinese postman. *Mathematical Programming*, 5 (1973) 88 – 124.
- [3] H.A. Eiselt, M. Gendreau, G. Laporte, Arc routing problems, part I: the Chinese postman problem. *Operations Research*, 43(2) (1995a) 231 – 242.
- [4] D. Goldfarb, J. Hao, S.R. Kai, Efficient Shortest Path Simplex Algorithms. *Operations Research*, 38(4) (1990) 624 – 628.
- [5] M. Grötschel, Z. Win, A cutting plane algorithm for the Wind Postman Problem, *Mathematical Programming*, 55 (1992) 339 – 358.
- [6] C.H. Kappauf, G.J. Koehler, The Mixed Postman Problem, *Disc. Applied Mathematics*, 1 (1979) 89 – 103.
- [7] G. Morales, Notas de Aula: Tópicos Avançados em Programação Linear, Relatório Técnico 08/2000, Laboratório de Engenharia de Produção, UENF, Campos – RJ, (2000) 45 p.
- [8] Y. Nobert, J.C. Picard, An optimal algorithm for the mixed Chinese postman problem. *Networks*, 27 (1996) 95 – 108.
- [9] F.G. Paes, Otimização de Rotas para Coleta do Lixo Doméstico: Um Tratamento GRASP do Problema do Carteiro Chinês Misto (PCCM). Dissertação de Mestrado em Ciências de Engenharia, UENF, Campos-RJ, 116 p., 2004.
- [10] W.L. Pearn, C.M. Liu, Algorithms for the Chinese postman problem on mixed networks. *Computers & Operations Research*, 22(5) (1995) 479 – 489.
- [11] W.L. Pearn, J.B. Chou, Improved solutions for the Chinese postman problem on mixed networks. *Computers & Operations Research*, 26 (1999) 819 – 827.
- [12] T.K. Ralphs, On the Chinese postman problem. *Operations Research Letters*. 14 (1993) 123 – 127.

This document was created with Win2PDF available at <http://www.daneprairie.com>.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.