UM ALGORITMO BRANCH-AND-CUT PARA O PROBLEMA DO CICLO DOMINANTE

Lucas Porto Maziero
Instituto de Computação
Universidade Estadual de Campinas (UNICAMP) Campinas - SP - Brasil
lucasporto1992@gmail.com

Fábio Luiz Usberti
Instituto de Computação
Universidade Estadual de Campinas (UNICAMP) Campinas - SP - Brasil
fusberti@ic.unicamp.br

Celso Cavellucci
Instituto de Computação
Universidade Estadual de Campinas (UNICAMP) Campinas - SP - Brasil
ccavellucci@gmail.com

RESUMO

Este artigo apresenta um estudo de uma generalização do Problema do Caixeiro Viajante (TSP), denominado Problema do Ciclo Dominante (DCP). Essa generalização consiste na composição de dois problemas NP-difíceis: o TSP e o Problema do Conjunto k-Dominante em Grafos (k-DSP). O objetivo do DCP consiste em encontrar um ciclo de custo mínimo em um grafo não-direcionado, partindo de um nó origem. O ciclo é visitado por um viajante que necessita visitar um conjunto de clientes (nós dominantes). A motivação de estudo do DCP consiste em sua aplicação prática para empresas de distribuição de energia, gás ou água; em particular, no processo de definição das rotas para a leitura dos consumos desses serviços. Neste artigo, é apresentada uma formulação de programação linear inteira e um algoritmo branch-and-cut para o DCP. Os resultados obtidos pelos métodos exatos são comparados com os resultados obtidos por uma metaheurística já desenvolvida para o DCP.

PALAVRAS CHAVE. Problema do Ciclo Dominante, Algoritmo Branch-and-Cut, Metaheurística.

ABSTRACT

This paper presents a study of a generalization of the Travelling Salesman Problem (TSP), called Dominant Cycle Problem (DCP). This generalization is the composition of two NP-hard problems: TSP and Set Problem k-Dominante in Graphs (k-DSP). The aim of the DCP is to find a minimum cost cycle in an undirected graph, from a source node. The cycle is trafficked by a traveler who needs to visit a set of customers (we dominant). The motivation of DCP study is in its practical application to power distribution companies, gas or water; in particular, the process of defining routes for reading the consumption of such services. In this paper, an integer linear programming formulation and branch-and-cut algorithm for the DCP are presented. The results obtained by exact methods are compared with the results obtained by metaheuristic already developed for the DCP.

KEYWORDS. Dominant Cycle Problem, Branch-and-Cut Algorithm, Metaheuristic.

1. Introdução

Neste trabalho, é investigado o Problema do Ciclo Dominante, do inglês *Dominant Cycle Problem* (DCP), que generaliza o Problema do Caixeiro Viajante, do inglês *Traveling Salesman Problem* (TSP) e o Problema do Conjunto k-Dominante em Grafos, do inglês *Dominating Set Problem* (k-DSP).

O TSP é definido do seguinte modo: dado um grafo completo e não-direcionado $G(V, E)$, onde V representa o conjunto de vértices e E o conjunto de arestas. Cada aresta $(i, j) \in E$ possui um custo não-negativo $c_{ij} \geqslant 0$ ([Papadimitriou e Steiglitz, 1998]). Para este trabalho, serão assumidos custos simétricos nas arestas, $c_{ij} = c_{ji}$. Seja a definição de ciclo hamiltoniano em G um ciclo que contém todos os vértices de G ([Bondy e Murty, 1976]). Então como solução o TSP busca encontrar, partindo de um vértice de origem, um ciclo hamiltoniano de custo mínimo que visita todos os vértices do grafo.

No k-DSP, é dado um grafo não-direcionado $G = (V, E)$ onde V é o conjunto dos vértices e E é o conjunto das arestas. Um vértice i é vizinho de um vértice j se e somente se $d(i, j) \leq k$, onde $d(i, j)$ é a distância (ou custo) de um caminho mínimo entre os nós i e j, enquanto k é denominado como o *raio de vizinhança*. Um subconjunto de vértices $D \subseteq V$ é um conjunto k-dominante de G se para cada vértice $v \in V \setminus D$ existe um vértice $u \in D$ tal que $d(u, v) \leq k$ [Chang, 1999].

Uma motivação para o estudo do DCP consiste em sua aplicação para um problema real de roteamento de leituristas. Empresas que atuam como fornecedores de energia elétrica, água e gás, possuem colaboradores responsáveis por registrar o consumo dos clientes dispersos geograficamente nas ruas e avenidas dos bairros de um centro urbano. O roteamento desses funcionários, denominados leituristas, gera um problema de difícil solução, conforme mostram [Usberti et al., 2008], [Usberti et al., 2011b], e [Usberti et al., 2011a], que consiste na elaboração de rotas nas quais os leituristas devem percorrer para realizar a leitura de consumo dos clientes.

Brasek [2004] e Jamil [2008] mostram que na última década, o avanço tecnológico introduziu os aparelhos de leitura remota , o que permite que um leiturista possa realizar um registro de consumo dentro de um raio de alcance máximo do aparelho. Nesse sentido, as rotas dos leituristas não precisam necessariamente visitar todos os clientes, contanto que todos os clientes estejam dentro do raio de alcance da rota. Uma solução ótima para esse problema real minimiza os tempos de percurso dos leituristas, de modo que todos os clientes do centro urbano tenham seus consumos registrados.

2. Problema do Ciclo Dominante

O DCP, introduzido e formulado inicialmente por [Current e Schilling, 1989], é definido a seguir. Considere um grafo completo e não-direcionado $G(V, E)$, onde V representa o conjunto de vértices e E o conjunto de arestas. Cada aresta $(i, j) \in E$ possui um custo não-negativo $c_{ij} \geqslant 0$. Serão considerados custos simétricos nas arestas, $c_{ij} = c_{ji}$. O vértice v_0 é um nó especial, denominado raiz. Um subconjunto de vértices $D \subseteq V$ é um conjunto k-dominante de G, se para cada vértice $v \in V \setminus D$, existe um vértice $u \in D$ tal que $d(v, u) \leqslant k$, onde $d(v, u)$ é a distância (ou custo) entre os nós v e u, enquanto k é o raio de vizinhança. Como solução, o DCP busca encontrar, partindo do vértice raiz, um ciclo de custo mínimo cujo vértices representam o conjunto k-dominante.

A Figura 1 ilustra uma solução viável para uma instância do DCP. Os pontos em cor vermelha representam o subconjunto de nós que dominam o grafo; o ponto em cor azul corresponde ao nó raiz; o traçado da rota encontra-se em cor preta; a vizinhança de um nó do conjunto k-dominante é demonstrada pelas circunferências em verde.

3. Formulação de Programação Linear Inteira

Uma formulação de Programação Linear Inteira (PLI) para o DCP é apresentada a seguir, onde a variável binária x_{ij} representa se um aresta $(i, j) \in E$ pertence (1) ou não (0) ao subciclo de

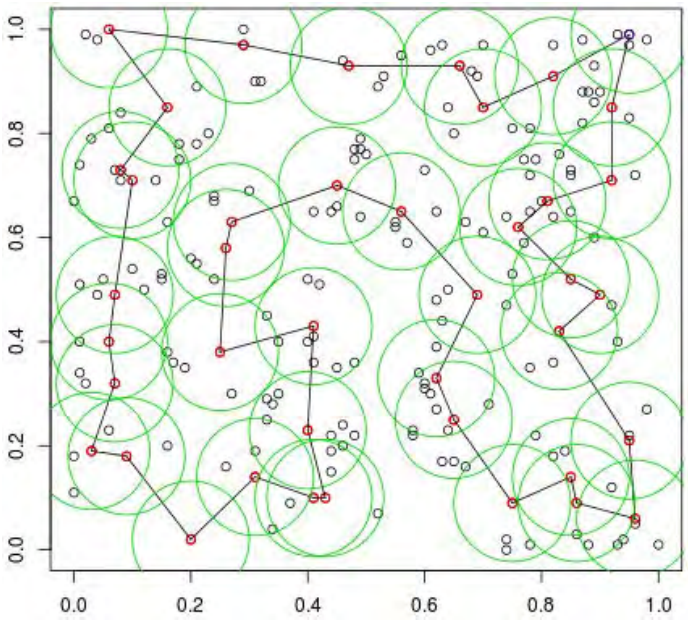

Figura 1: Solução factível para o DCP, com 200 nós e raio de vizinhança $k = 0.10$.

custo mínimo e a variável binária y_i revela quando um nó pertence (1) ou não (0) ao ciclo. S é um subconjunto de vértices de G e $\delta(v)$ corresponde ao conjunto de nós adjacentes ao vértice v.

$$MIN \quad \sum_{(i,j)\in E} c_{ij}x_{ij} \tag{1}$$

s.a.

$$y_0 = 1 \tag{2}$$

$$\sum_{j\in N_k(i)} y_j \geqslant 1 \qquad \forall i \in V \tag{3}$$

$$\sum_{i\in\delta(j)} x_{ij} = y_j, \qquad \forall j \in V \tag{4}$$

$$\sum_{j\in\delta(i)} x_{ij} = y_i, \qquad \forall i \in V \tag{5}$$

$$\sum_{i,j\in S} x_{ij} \leq |S| - 1, \qquad \forall S \subseteq V\backslash\{v_0\} \tag{6}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i,j \in V \tag{7}$$

$$y_i \in \{0,1\} \qquad \forall i \in V \tag{8}$$

A função objetivo (1) busca por um subciclo de custo mínimo, onde o custo de uma solução é dado pela somatória dos custos de suas arestas. A restrição (2) assegura que o vértice raiz pertence ao conjunto k-dominante. A restrição (3) garante que todo vértice deve ser vizinho de pelo menos um elemento do conjunto k-dominante. O conjunto de restrições (4) e (5) garantem que um vértice será visitado pelo ciclo quando o mesmo pertencer ao conjunto k-dominante. A restrição (6) impede a ocorrência de subciclos inválidos entre os vértices, ou seja, evita os subciclos que não incidem no vértice raiz.

Seja (x', y') uma solução inteira para a formulação DCP sem as restrições (6). Um algoritmo para separação da Desigualdade 6 é proposto: como pré-processamento, uma busca em

profundidade é realizada no subgrafo induzido pela solução (x', y'), com objetivo de identificar os subciclos ilegais. Seja C um subciclo ilegal identificado pela busca em profundidade na solução (x', y'). O algoritmo de separação consiste em preencher um subconjunto S com todos os vértices que compõem o subciclo C. Após isso, adiciona-se ao modelo DCP a Desigualdade 6 associada ao subconjunto S. O algoritmo itera o mesmo procedimento para cada subciclo ilegal identificado na solução (x', y').

4. Algoritmo *Branch-and-Cut*

4.1. Desigualdade Válida

Considere a desigualdade (9):

$$\sum_{(i,j)\in S} x_{ij} \leq |S| - 1, \qquad \forall S \subseteq V\backslash\{v_0\} \tag{9}$$

A desigualdade (9) faz parte do modelo matemático para o DCP apresentado no Capítulo 3. Trata-se de uma restrição que impede a ocorrência de subciclos ilegais, ou seja, subciclos que não incidem no vértice raiz (v_0).

Com o objetivo de fortalecer a desigualdade (9), a expressão $|S| - 1$ é substituída por $\sum_{i\in S} y_i - 1$:

$$\sum_{(i,j)\in S} x_{ij} \leq \sum_{i\in S} y_i - 1, \qquad \forall S \subseteq V\backslash\{v_0\} \tag{10}$$

No trabalho de [Laporte, 1986], são apresentadas famílias de desigualdades válidas obtidas através de generalizações das restrições originais da formulação para o TSP. Essas desigualdades válidas são semelhantes a desigualdade (10) no que diz respeito a eliminação de subciclos ilegais.

Existem subconjuntos de vértices específicos onde a desigualdade (10) torna-se inválida. Supondo que um subconjunto de vértices S não possui vértices presentes no conjunto k-dominante, ou seja, $\sum_{i\in S} y_i = 0$, então a expressão $\sum_{i\in S} y_i - 1$ resultará em valor negativo (-1), tornando a desigualdade (10) inválida pois o número de arestas presentes em S deve ser limitado pelo número de vértices do conjunto k-dominante em S.

Para que a expressão $\sum_{i\in S} y_i - 1$ seja utilizada, uma nova família de subconjuntos de vértices é definida. Seja $\gamma(V) = \{F \subseteq V\backslash\{v_0\} : \exists v \in F, N_k(v) \subseteq F\}$, onde $N_k(v)$ representa a vizinhança de v. $\gamma(V)$ é a família de todos os subconjuntos de vértices que contém a vizinhança de pelo menos um vértice $v \in V$. Portanto, dado um subconjunto de vértices $S \in \gamma(V)$, pelo menos um vértice $v \in S$ irá pertencer ao conjunto k-dominante, validando a expressão $\sum_{i\in S} y_i - 1$.

Utilizando a definição de $\gamma(V)$, definimos uma desigualdade de vizinhança válida para o DCP:

$$\sum_{(i,j)\in S} x_{ij} \leq \sum_{i\in S} y_i - 1, \qquad \forall S \subseteq \gamma(V) \tag{11}$$

A desigualdade (11) também impede a ocorrência de subciclos ilegais, pois o número de arestas presentes em S deve ser limitado pelo número de vértices do conjunto k-dominante em S.

É possível observar que a desigualdade (11) é mais forte que a desigualdade (9) para subconjuntos de vértices $S \subseteq \gamma(V)$ onde pelo menos um vértice $i \in S$ não pertence ao conjunto k-dominante. Nesse caso, o número de vértices presentes no conjunto k-dominante em S é necessariamente menor que a cardinalidade de S. Portanto, $\sum_{i\in S} y_i - 1 < |S| - 1$.

4.2. Rotinas de Separação

4.2.1. Vizinhanças Eficazes

Existem subconjuntos de vértices particulares em $\gamma(V)$ que tornam uma vizinhança $N_k(v)$ eficaz ou ineficaz para a Desigualdade 11 efetivar o corte. Seja $S \subseteq \gamma(V)$ um subconjunto de vértices com n subciclos ilegais formados. Considerando uma vizinhança $N_k(v)$ para algum vértice $v \in S$, $N_k(v)$ é eficaz se todos os seus vértices correspondentes que são do conjunto k-dominante pertencem a algum subciclo ilegal formado em S, caso contrário, $N_k(v)$ é uma vizinhança completa ineficaz. A Desigualdade 11 impede a ocorrência de subciclos ilegais somente se uma vizinhança completa eficaz for encontrada em S, pois o número de arestas presentes na solução em S deve ser limitado pela quantidade de vértices do conjunto k-dominante, que estão presentes em algum dos n subciclos ilegais formados em S.

Supondo um subconjunto de vértices $S = \{6, 7, 8, 9\}$, $N_k(7) = \{6, 7\}$ e o subciclo inválido formado pelos vértices $\{7, 8, 9\}$. Desigualdade 11 será constituída da seguinte maneira:

$$x_{(6,7)} + x_{(6,8)} + x_{(6,9)} + x_{(7,8)} + x_{(7,9)} + x_{(8,9)} \leq y_6 + y_7 + y_8 + y_9 - 1 =$$
$$0 + 0 + 0 + 1 + 1 + 1 \leq 0 + 1 + 1 + 1 - 1 =$$
$$3 \leq 2$$

Então, pela Desigualdade 11 a solução onde o subciclo inválido $\{7, 8, 9\}$ se encontra será cortada.

Duas rotinas de separação para a Desigualdade (11) foram implementadas. A primeira rotina considera todas as vizinhanças completas eficazes em um subconjunto de vértices. A segunda rotina visa somente a vizinhança completa eficaz de cardinalidade mínima.

Seja (x', y') uma solução inteira para a formulação DCP sem as restrições de eliminação de subciclos ilegais (9). Como pré-processamento para as duas rotinas de separação, uma busca em profundidade é realizada no subgrafo induzido pela solução inteira (x', y') obtida no processo de solução da formulação DCP. A busca em profundidade tem por objetivo identificar os subciclos ilegais.

4.2.2. Múltiplas Vizinhanças

Dados os n subciclos ilegais encontrados pela busca em profundidade, o conjunto S será formado pelos vértices que compõem o n-ésimo subciclo ilegal. O primeiro passo é verificar para cada vértice v que está no conjunto S, se sua vizinhança completa está contida em S, ou seja, se o conjunto de vértices correspondentes a vizinhança do vértice v compõem o n-ésimo subciclo ilegal. Se uma vizinhança completa for encontrada em S, a Desigualdade 11 será adicionada ao modelo DCP. Caso contrário, é necessário verificar para cada vértice v que está no n-ésimo subciclo ilegal, se sua vizinhança completa é eficaz para a Desigualdade 11. Nesse segundo passo, o conjunto S será construído para cada vértice v, onde sua composição será dada pelos vértices que estão no n-ésimo subciclo ilegal e pelos vértices correspondentes a vizinhança completa do vértice v. Se a vizinhança completa do vértice v for eficaz, será adicionada ao modelo DCP a Desigualdade 11 com o respectivo conjunto S. Esse segundo passo é repetido para todos os vértices que estão no n-ésimo subciclo ilegal.

Se o primeiro e o segundo passo da rotina de separação não encontrarem Desigualdades 11 violadas, o terceiro passo será responsável por adicionar ao modelo DCP a Desigualdade 9.

A rotina de separação repetirá os passos até que todos os n subciclos ilegais sejam avaliados. O Pseudocódigo 1 ilustra o procedimento.

4.2.3. Vizinhança de Cardinalidade Mínima

A segunda rotina de separação implementada difere em relação a primeira somente no segundo passo. Ao invés de considerar todas as vizinhanças completas eficazes e adicioná-las ao

modelo DCP, essa rotina visa a vizinhança completa eficaz de cardinalidade mínima. Da mesma maneira que na primeira rotina implementada, é necessário verificar para cada vértice v que está no n-ésimo subciclo ilegal, se sua vizinhança completa é eficaz para a Desigualdade 11. O conjunto S será construído para cada vértice v, onde sua composição será dada pelos vértices que estão no n-ésimo subciclo ilegal e pelos vértices correspondentes a vizinhança completa do vértice v. Uma vizinhança completa eficaz de algum vértice v é considerada mínima para o n-ésimo subciclo ilegal, se o número de vértices correspondentes a vizinhança do vértice v que não estão no conjunto k-dominante D é mínimo dentre todas as vizinhanças completas eficazes dos vértices que estão no n-ésimo subciclo ilegal. Será adicionada ao modelo DCP a Desigualdade 11 cujo o conjunto S representa a vizinhança completa eficaz de cardinalidade mínima.

A rotina de separação repetirá os passos até que todos os n subciclos ilegais sejam avaliados. O Pseudocódigo 2 ilustra o procedimento.

5. Experimentos Computacionais

Os resultados obtidos consistem em experimentos computacionais sobre o modelos matemático para o DCP e sobre o Algoritmo *Branch-and-Cut*. Também faz parte dos resultados experimentos computacionais realizados com a metaheurística GRASP proposta por [Maziero et al., 2015]. As instâncias utilizadas nos experimentos computacionais foram geradas distribuindo-se pontos (vértices) em um plano de dimensões 1×1 com coordenadas (x, y), geradas aleatoriamente no intervalo $[0, 1]$ com distribuição uniforme. As instâncias são categorizadas pelo par (n, k), tal que n é o número de nós e k é o raio de vizinhança. Foram geradas instâncias com valores de $n = \{50, 100, 200, 400, 800\}$ e $k = \{0; 0.02; 0.04; 0.06; 0.08; 0.1\}$.

O *solver* utilizado para a solução do modelo DCP e para o algoritmo *Branch-and-Cut* foi o Gurobi[1], configurado com um tempo de execução máximo de 60 minutos. A metaheurística GRASP foi implementada na linguagem de programação C++ e compilada com g++ versão 4.7.2, configurada também com um tempo de execução máximo de 60 minutos. Os experimentos computacionais foram executados em um computador com sistema operacional Ubuntu versão 12.10 e processador Intel Xeon CPU X3430 2,40 GHz, com 8GB de memória RAM.

As Tabelas 1, 2 e 3 reportam os resultados obtidos, onde a Tabela 1 apresenta os resultados obtidos pelo modelo para o DCP, a Tabela 2 os resultados obtidos pelo Algoritmo *Branch-and-Cut* utilizando a rotina de separação de *Múltiplas Vizinhanças*, e a Tabela 3 os resultados obtidos pelo Algoritmo *Branch-and-Cut* utilizando a rotina de separação de *Vizinhança de Cardinalidade Mínima*. Em todos os métodos exatos a melhor solução primal obtida pela metaheurística GRASP é transmitida ao *solver*, estabelecendo um limitante superior inicial.

Nas Tabelas 1, 2 e 3 os resultados obtidos dos experimentos computacionais pela metodologia exata são comparados com os resultados obtidos dos experimentos computacionais pela metaheurística GRASP, organizando-os por cada tipo de instâncias, definidas pelo par (n, k). Em relação aos resultados obtidos pela metodologia exata, são apresentados os tamanhos dos conjuntos k-dominantes ($|D|m$), os limitantes primais ($LP2$) e duais (LD), o desvio de otimalidade correspondente ($GAP(\%)lp2 = 100 * ((LP2 - LD)/LP2)$) e os tempos de execução ($Tempo$). Com relação aos resultados obtidos pela metaheurística GRASP, são apresentados os tamanhos dos conjuntos k-dominantes ($|D|g$), o limitantes primais ($LP1$) alcançados e o desvio de otimalidade correspondente ($GAP(\%)lp1 = 100 * ((LP1 - LD)/LP1)$).

Os resultados das Tabelas 1, 2 e 3 demonstram que o *solver* atingiu soluções ótimas em 15 dos 30 tipos de (n, k) em cada metodologia exata. Dos 30 tipos de (n, k), a metaheurística GRASP conseguiu atingir 8 soluções ótimas. Nas instâncias onde se desconhece a solução ótima, a metodologia exata através do modelo matemático para o DCP apresentou elevados desvios de

[1]O Gurobi é um solver para problemas de programação matemática. Para otimização de problemas de programação inteira e programação inteira mista, o Gurobi utiliza o algoritmo branch-and-bound em conjunto com técnicas de plano de corte e heurísticas (Gurobi Optimization Inc., 2015).

Algorithm 1: Rotina de Separação - Múltiplas Vizinhanças

```
início
    para i de 1 até n subciclos ilegais faça
        S = vértices do subciclo ilegal i;
        vizinhancaCompletaNoCiclo = 0;
        para j de 1 até v vértices do subciclo ileal i faça
            se N_k(j) está contida no subciclo ilegal i então
                Adicione ao modelo DCP a Desigualdade 11 com o conjunto S;
                vizinhancaCompletaNoCiclo = 1;
                Interrompa o laço;
        se vizinhancaCompletaNoCiclo == 0 então
            vizinhancaCompleta = 0;
            para j de 1 até v vértices do subciclo ileal i faça
                se N_k(j) é uma vizinhança completa eficaz para a Desigualdade 11
                então
                    S = vértices do subciclo ilegal i ∪N_k(j);
                    Adicione ao modelo DCP a Desigualdade 11 com o conjunto S;
                    vizinhancaCompleta = 1;
        se vizinhancaCompletaNoCiclo == 0 e vizinhancaCompleta == 0 então
            Adicione ao modelo DCP a Desigualdade 9;
fim
```

Algorithm 2: Rotina de Separação - Vizinhança de Cardinalidade Mínima

```
início
    para i de 1 até n subciclos ilegais faça
        S = vértices do subciclo ilegal i;
        vizinhancaCompletaNoCiclo = 0;
        para j de 1 até v vértices do subciclo ileal i faça
            se N_k(j) está contida no subciclo ilegal i então
                Adicione ao modelo DCP a Desigualdade 11 com o conjunto S;
                vizinhancaCompletaNoCiclo = 1;
                Interrompa o laço;
        se vizinhancaCompletaNoCiclo == 0 então
            vizinhancaCompleta = 0;
            menorVizinhanca = infinito;
            menorS;
            para j de 1 até v vértices do subciclo ileal i faça
                se N_k(j) é uma vizinhança completa eficaz para a Desigualdade 11
                então
                    S = vértices do subciclo ilegal i ∪N_k(j);
                    se Número de vértices v ∈ N_k(j) \ D < menorVizinhanca então
                        menorVizinhanca = Número de vértices v ∈ N_k(j) \ D;
                        menorS = S;
            Adicione ao modelo DCP a Desigualdade 11 com o conjunto menorS;
            vizinhancaCompleta = 1;
        se vizinhancaCompletaNoCiclo == 0 e vizinhancaCompleta == 0 então
            Adicione ao modelo DCP a Desigualdade 9;
fim
```

otimalidade, quando comparados aos desvios de otimalidade obtidos pelos Algoritmos *Branch-and-Cut*. Em todas dessas instâncias, os desvios de otimalidade obtidos pelo Algoritmo *Branch-and-Cut* que implementa a rotina de separação *Vizinhança de Cardinalidade Mínima*, foram menores aos desvios de otimalidade obtidos pelo modelo para o DCP; já os desvios de otimalidade obtidos pelo Algoritmo *Branch-and-Cut* que implementa a rotina de separação *Múltiplas Vizinhanças*, foram menores aos desvios de otimalidade obtidos pelo modelo para o DCP com exceção da instância $n = 400$ nós e $k = 0.02$ de raio de vizinhança.

Além de comparar os desvios de otimalidade obtidos por cada metodologia exata, é possível também analisar esses desvios de otimalidade de maneira mais ampla usando um gráfico de caixa de trama, apresentado na Figura 2. O gráfico fornece informações de desvios de otimalidade obtidos pelas metodologias exatas e pela metaheurística GRASP, para todas as instâncias definidas pelos pares (n, k). Em cor azul, verde e roxa se encontram os desvios de otimalidade alcançados pela metaheurística GRASP em relação a cada metodologia exata. Em cor laranja, vermelha e marrom se encontram os desvios de otimalidade alcançados pelo modelo matemático para o DCP, pelo Algoritmo *Branch-and-Cut* que implementa a rotina de separação *Múltiplas Vizinhanças*, e pelo Algoritmo *Branch-and-Cut* que implementa a rotina de separação *Vizinhança de Cardinalidade Mínima*, respectivamente. Em geral, os desvios obtidos pela metaheurística GRASP apresentam uma variação maior quando comparados aos desvios obtidos pelas metodologias exatas.

Analisando os desvios obtidos pela metaheurística GRASP em relação aos desvios obtidos pelas metodologias exatas, o primeiro quartil das metodologias exatas é sempre menor ou igual em relação ao primeiro quartil da metaheurística GRASP, sendo o mesmo fato válido para as medianas. Isso demonstra que as metologias exatas foram eficazes em encontrar melhores limitantes, mesmo recebendo como limitante superior inicial a melhor solução primal obtida pela metaheurística GRASP.

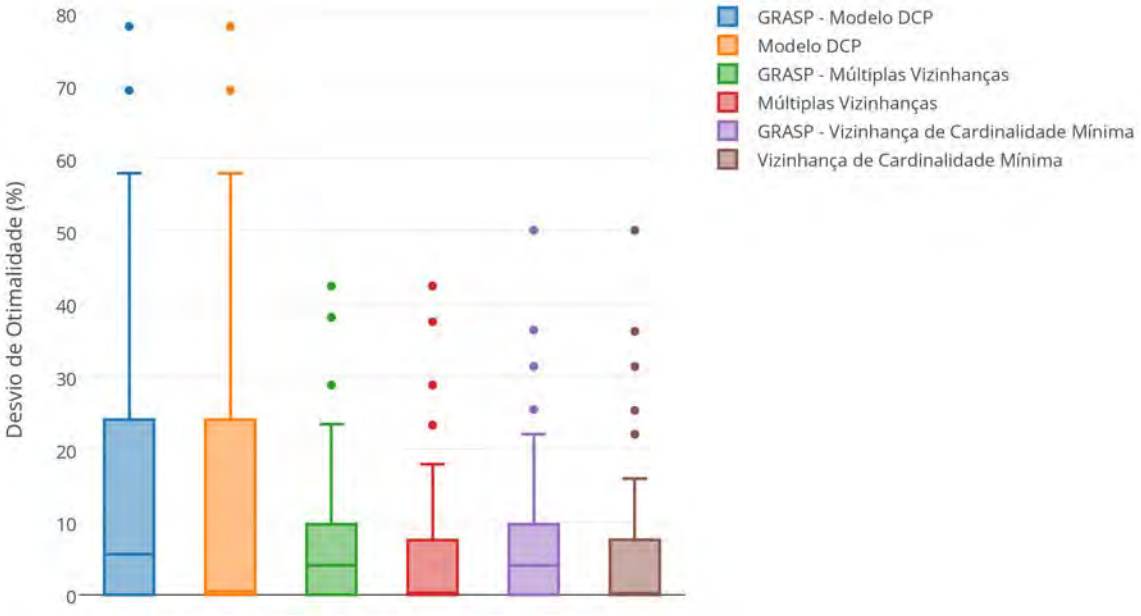

Figura 2: Distribuição do Desvio de Otimalidade (%) dos métodos exatos.

6. Conclusões

Os resultados obtidos pelos experimentos computacionais através dos métodos exatos demonstraram a eficácia do Algoritmo *Branch-and-Cut*, com suas diferentes rotinas de separação, em encontrar bons limitantes duas para o DCP. Para as instâncias com 50 e 100 nós, todos os métodos obtiveram o mesmo desvio de otimalidade. Nas demais instâncias, o Algoritmo *Branch-and-Cut* que implementa a rotina de separação *Múltiplas Vizinhanças*, e o Algoritmo *Branch-and-Cut* que

implementa a rotina de separação *Vizinhança de Cardinalidade Mínima*, foram notoriamente melhores que o modelo matemático para o DCP. Ambos os algoritmos tiveram desempenho parecidos ao obterem seus desvios, porém o Algoritmo *Branch-and-Cut* que implementa a rotina de separação *Múltiplas Vizinhanças* obteve uma variação maior em seus desvios de otimalidade.

Em relação a qualidade das soluções primais obtidas pelos métodos exatos, para as instâncias com 50 e 100 nós, todos os métodos obtiveram o mesmo custo. Nas demais instâncias, houve pequenas variações entre os métodos.

Novas investigações referentes a metodologias de solução para o DCP são interessantes. Outros métodos heurísticos e exatos podem serem estudados e desenvolvidos com o intuito de obter melhores limitantes para o DCP. Além disso, a criação de novas instâncias e a aplicação das metodologias apresentadas neste trabalho a cenários reais envolvendo o problema de roteamento de leituristas, irão contribuir com novos resultados para o DCP.

7. Agradecimentos

Este trabalho teve o suporte de FAEPEX-UNICAMP, CAPES e FAPESP. Agradecemos aos revisores anônimos pelos comentários.

Referências

Bondy, J. A. e Murty, U. S. R. (1976). *Graph theory with applications*, volume 290. Macmillan London.

Brasek, C. (2004). Urban utilities warm up to the idea of wireless automatic meter reading. *Computing Control Engineering Journal*, 15(6):10–14.

Chang, G. J. (1999). Algorithmic aspects of domination in graphs. In *Handbook of combinatorial optimization*, p. 1811–1877. Springer.

Current, J. R. e Schilling, D. A. (1989). The covering salesman problem. *Transportation science*, 23(3):208–213.

Jamil, T. (2008). Design and implementation of a wireless automatic meter reading system. In *Proceedings of the World Congress on Engineering 2008 – WCE 2008*.

Laporte, G. (1986). Generalized subtour elimination constraints and connectivity constraints. *Journal of the Operational Research Society*, p. 509–514.

Maziero, L. P., Usberti, F. L., e Cavellucci, C. (2015). O problema do ciclo dominante. *SBPO XLVII Proceedings–Brazilian Symposium of Operations Research*.

Papadimitriou, C. H. e Steiglitz, K. (1998). *Combinatorial optimization: algorithms and complexity*. Courier Corporation.

Usberti, F. L., França, P. M., e França, A. L. M. (2008). *Roteamento de Leituristas: Um Problema NP-Difícil (in portuguese). In: XL SBPO Brazilian Symposium of Operational Research*. Annals XL SBPO, João Pessoa.

Usberti, F. L., França, P. M., e França, A. L. M. (2011a). Grasp with evolutionary path-relinking for the capacitated arc routing problem. *Computers and Operations Research*. ISSN 0305-0548. doi: 10.1016/j.cor.2011.10.014.

Usberti, F. L., França, P. M., e França, A. L. M. (2011b). The open capacitated arc routing problem. *Computers and Operations Research*, 38(11):1543 – 1555. ISSN 0305-0548.

Tabela 1: Resultados dos experimentos computacionais através do modelo para o DCP

		DCP							
n	k	$\|D\|g$	$LP1$	$\|D\|m$	$LP2$	LD	$GAP(\%)lp1$	$GAP(\%)lp2$	$Tempo$
50	0,10	29	5,13	29	5,13	5,13	0,00	0,00	0
	0,08	38	5,59	38	5,59	5,59	0,00	0,00	0
	0,06	41	5,78	41	5,78	5,78	0,00	0,00	0
	0,04	44	5,84	44	5,84	5,84	0,00	0,00	0
	0,02	47	5,92	47	5,92	5,92	0,00	0,00	0
	0,00	50	5,96	50	5,96	5,96	0,00	0,00	0
100	0,10	37	5,34	37	5,34	5,34	0,00	0,00	274
	0,08	51	5,9	51	5,90	5,90	0,00	0,00	441
	0,06	65	7,02	65	6,99	6,99	0,43	0,00	61
	0,04	87	7,8	87	7,73	7,73	0,90	0,00	0
	0,02	97	7,96	97	7,94	7,94	0,25	0,00	0
	0,00	100	7,98	100	7,97	7,97	0,13	0,00	0
200	0,10	39	5,5	39	5,50	4,02	26,91	26,91	3600
	0,08	51	6,4	53	6,30	5,33	16,72	15,40	3600
	0,06	75	7,68	79	7,62	6,86	10,68	9,97	3600
	0,04	116	9,19	117	8,95	8,81	4,13	1,56	3600
	0,02	176	10,54	180	10,31	10,08	4,36	2,23	3600
	0,00	200	10,79	200	10,36	10,36	3,99	0,00	17
400	0,10	46	5,45	46	5,45	2,29	57,98	57,98	3600
	0,08	66	7,02	66	7,02	4,08	41,88	41,88	3600
	0,06	104	8,67	104	8,67	6,45	25,61	25,61	3600
	0,04	189	11,9	192	11,49	10,92	8,24	4,96	3600
	0,02	345	15,66	349	14,62	14,48	7,54	0,96	3600
	0,00	400	16,15	400	15,07	15,07	6,69	0,00	127
800	0,10	54	5,87	54	5,87	1,28	78,19	78,19	3600
	0,08	76	7,41	76	7,41	2,27	69,37	69,37	3600
	0,06	114	9,39	114	9,39	4,31	54,10	54,10	3600
	0,04	224	13,67	224	13,67	10,38	24,07	24,07	3600
	0,02	559	20,31	559	20,31	18,65	8,17	8,17	3600
	0,00	800	23,45	800	21,18	21,18	9,68	0,00	581

n – número de nós. k – raio de vizinhança.

Tabela 2: Resultados dos experimentos computacionais através do Algoritmo *Branch-and-Cut*, utilizando a rotina de separação de *Múltiplas Vizinhanças*

		Múltiplas Vizinhanças							
n	k	$\|D\|g$	$LP1$	$\|D\|m$	$LP2$	LD	$GAP(\%)lp1$	$GAP(\%)lp2$	$Tempo$
50	0,10	29	5,13	29	5,13	5,13	0,00	0,00	0
	0,08	38	5,59	38	5,59	5,59	0,00	0,00	0
	0,06	41	5,78	41	5,78	5,78	0,00	0,00	0
	0,04	44	5,84	44	5,84	5,84	0,00	0,00	0
	0,02	47	5,92	47	5,92	5,92	0,00	0,00	0
	0,00	50	5,96	50	5,96	5,96	0,00	0,00	0
100	0,10	37	5,34	37	5,34	5,34	0,00	0,00	14
	0,08	51	5,9	51	5,90	5,90	0,00	0,00	212
	0,06	65	7,02	65	6,99	6,99	0,43	0,00	16
	0,04	87	7,8	87	7,73	7,73	0,90	0,00	0
	0,02	97	7,96	97	7,94	7,94	0,25	0,00	0
	0,00	100	7,98	100	7,97	7,97	0,13	0,00	0
200	0,10	39	5,5	39	5,50	5,14	6,55	6,55	3600
	0,08	51	6,4	56	6,39	6,13	4,22	4,07	3600
	0,06	75	7,68	75	7,65	7,42	3,39	3,01	3600
	0,04	116	9,19	119	8,92	8,88	3,37	0,45	3600
	0,02	176	10,54	176	10,18	10,11	4,08	0,69	3600
	0,00	200	10,79	200	10,36	10,36	3,99	0,00	17
400	0,10	46	5,45	47	5,44	4,17	23,49	23,35	3600
	0,08	66	7,02	66	7,02	5,76	17,95	17,95	3600
	0,06	104	8,67	106	8,65	7,53	13,15	12,95	3600
	0,04	189	11,9	195	11,35	11,17	6,13	1,59	3600
	0,02	345	15,66	345	15,65	14,50	7,41	7,35	3600
	0,00	400	16,15	400	15,07	15,07	6,69	0,00	127
800	0,10	54	5,87	54	5,87	3,38	42,42	42,42	3600
	0,08	76	7,41	78	7,34	4,58	38,19	37,60	3600
	0,06	114	9,39	114	9,39	6,68	28,86	28,86	3600
	0,04	224	13,67	224	13,65	11,47	16,09	15,97	3600
	0,02	559	20,31	559	20,31	18,79	7,48	7,48	3600
	0,00	800	23,45	800	21,18	21,18	9,68	0,00	581

n – número de nós. k – raio de vizinhança.

Tabela 3: Resultados dos experimentos computacionais através do Algoritmo *Branch-and-Cut*, utilizando a rotina de separação de *Vizinhança de Cardinalidade Mínima*

		Vizinhança de Cardinalidade Mínima							
n	k	$\|D\|g$	$LP1$	$\|D\|m$	$LP2$	LD	$GAP(\%)lp1$	$GAP(\%)lp2$	$Tempo$
50	0,10	29	5,13	29	5,13	5,13	0,00	0,00	0
	0,08	38	5,59	38	5,59	5,59	0,00	0,00	0
	0,06	41	5,78	41	5,78	5,78	0,00	0,00	0
	0,04	44	5,84	44	5,84	5,84	0,00	0,00	0
	0,02	47	5,92	47	5,92	5,92	0,00	0,00	0
	0,00	50	5,96	50	5,96	5,96	0,00	0,00	0
100	0,10	37	5,34	37	5,34	5,34	0,00	0,00	14
	0,08	51	5,9	51	5,90	5,90	0,00	0,00	192
	0,06	65	7,02	65	6,99	6,99	0,43	0,00	17
	0,04	87	7,8	87	7,73	7,73	0,90	0,00	0
	0,02	97	7,96	97	7,94	7,94	0,25	0,00	0
	0,00	100	7,98	100	7,97	7,97	0,13	0,00	0
200	0,10	39	5,5	39	5,50	5,09	7,45	7,45	3600
	0,08	51	6,4	53	6,33	6,09	4,84	3,79	3600
	0,06	75	7,68	75	7,64	7,43	3,26	2,75	3600
	0,04	116	9,19	118	8,92	8,88	3,37	0,45	3600
	0,02	176	10,54	176	10,18	10,11	4,08	0,69	3600
	0,00	200	10,79	200	10,36	10,36	3,99	0,00	17
400	0,10	46	5,45	47	5,44	4,06	25,50	25,37	3600
	0,08	66	7,02	66	7,02	5,47	22,08	22,08	3600
	0,06	104	8,67	104	8,67	7,37	14,99	14,99	3600
	0,04	189	11,9	191	11,27	11,18	6,05	0,80	3600
	0,02	345	15,66	346	14,53	14,52	7,28	0,07	3600
	0,00	400	16,15	400	15,07	15,07	6,69	0,00	127
800	0,10	54	5,87	54	5,87	2,93	50,09	50,09	3600
	0,08	76	7,41	78	7,39	4,71	36,44	36,27	3600
	0,06	114	9,39	114	9,39	6,44	31,42	31,42	3600
	0,04	224	13,67	224	13,67	11,49	15,95	15,95	3600
	0,02	559	20,31	559	20,31	18,78	7,53	7,53	3600
	0,00	800	23,45	800	21,18	21,18	9,68	0,00	581

n – número de nós. k – raio de vizinhança.