PERFORMANCE DE DOIS *SOLVERS* NA RESOLUÇÃO DO PROBLEMA DE TRANSPORTE COM CUSTO LINEAR POR PARTES

Carise Elisane Schmidt
Universidade Federal do Paraná - UFPR
Centro Politécnico, Caixa Postal 19.011, CEP: 81.531-980, Curitiba, PR
carise.schmidt@ifsc.edu.br

Crisiane Rezende Vilela de Oliveira
Instituto Federal do Paraná - IFPR
Rua João Negrão, 1285, Rebouças, CEP: 80.230-150, Curitiba, PR
crisiane.oliveira@ifpr.edu.br

Arinei Carlos Lindbeck da Silva
Universidade Federal do Paraná - UFPR
Centro Politécnico, Caixa Postal 19.011, CEP: 81.531-980, Curitiba, PR
arinei@ufpr.br

RESUMO

Este trabalho apresenta um comparativo entre o desempenho dos *solvers* Cplex e Gurobi na resolução do Problema de Transporte Linear por Partes. Esse desempenho é importante para uma futura avaliação de outras metodologias de resolução, uma vez que esse é um problema NP-Hard. Para os testes, 1080 instâncias foram geradas aleatoriamente e divididas conforme o tamanho do problema e o número de modos no arco. O desempenho dos *solvers* foi avaliado com base nos valores da função objetivo e do *gap*, limitando o tempo de processamento em 120 segundos. O CPLEX mostrou-se vantajoso para problemas em que o valor ótimo foi atingido dentro do limite de tempo estabelecido. Nos problemas maiores e, especialmente para problemas quadrados, a performance do Gurobi, em relação a função objetivo, foi superior. Esse estudo apontou que, dentro das condições estabelecidas, o Gurobi apresentou melhor desempenho na resolução desse tipo de problema.

PALAVRAS CHAVE. Problema de transporte. Custo linear por partes. Solver de otimização.

Tópicos: PM - Programação Matemática; OC - Otimização Combinatória; L&T - Logística e Transportes.

ABSTRACT

In this paper we present a comparasion between the performance of the solvers Cplex and Gurobi in resolution of the Piecewise Linear Transportation Problem. That performance is important for a future valuation of others resolution methodologies, once the problem is NP-Hard. For the tests, we generated 1080 instances randomly and we dividided them acoording the size of the problems and the numbers of modes in the arc. We valuate the solver's performance based on the values of the objective function and gap, and we limited the processing time in 120 seconds. Cplex demonstrated to be useful for problems which the optimum value was found within the established time limit. For larger problems, and specially for square problems, Gurobi's performance, concerning the objective function, was higher. This study indicated that, inside the established conditions, the Gurobi show better performance in resolution this type of problem.

KEYWORDS. Tranportation problem. Piecewise linear cost. Otimization solver.

Paper topics: MP - Mathematical Programming; CO - Combinatorial Optimization; L&T - Logistics and Transport.

1. Introdução

Esse trabalho aborda a comparação entre o desempenho de dois *solvers* na resolução de problemas de transporte que apresentam uma estrutura de custo linear por partes. Por ser um problema do tipo *NP-Hard* [Sheng et al., 2006], o esforço computacional para a sua resolução cresce exponencialmente a medida que o tamanho do problema aumenta. Assim, embora algoritmos do tipo *branch-and-bound* ou *branch-and-cut* consigam encontrar soluções exatas para problemas de programação inteira-mista, esses métodos são geralmente ineficientes e computacionalmente caros quando aplicados a problemas de grandes dimensões [Sheng et al., 2006], não sendo possível evitar o crescimento exponencial do tempo de resolução e nem a necessidade de memória computacional para executá-los [Nemhauser e Wolsey, 1988].

Além disso, em situações práticas, o tempo requerido para alcançar uma solução exata é, frequentemente, inviável no contexto de uma empresa. Dessa forma, encontrar soluções aproximadas, dentro de um intervalo de tempo aceitável, pode ser a melhor opção para atender as necessidades da empresa, de forma satisfatória.

Diante desse contexto, avaliar outras metodologias de resolução, além das exatas, torna-se importante. Pensando em uma futura aplicação no desenvolvimento de um método híbrido para a resolução do problema de transporte linear por partes, o objetivo desse trabalho foi comparar o desempenho dos *solvers* Gurobi Optimizer 7.0.1 e IBM ILOG Cplex Interactive 12.6.3 na resolução desse tipo de problema.

A sequência do artigo é organizada conforme segue. Na Secção 2, o problema de transporte linear por partes é apresentado; a Seção 3 traz a definição formal e a formulação matemática do problema, que utiliza o modelo de múltipla escolha para descrever a estrutura linear por partes do custo. A Seção 4 especifica os procedimentos adotados na geração das instâncias de teste e na comparação de desempenho dos dois *solvers* para a resolução do PTLP. Na Seção 5 são expostos os resultados obtidos com a realização dos testes computacionais, e na Seção 6 são apresentadas as considerações finais.

2. O problema de transporte linear por partes

O Problema de Transporte (PT) é um dos típicos problemas abordados pela pesquisa operacional. Segundo [Silva, 2012], sua formulação já é conhecida desde 1941, quando foi apresentada por Hitchcock. Em 1954, uma generalização do problema de transporte foi proposta [Hirsch e Dantzig, 1954], através da inclusão de um custo fixo à estrutura já existente, denominado problema de transporte com custo fixo (PTCF). Esse problema também representa um caso particular dos Problemas Gerais de Custo Fixo (PGCF), os quais compõem uma classe bem conhecida de problemas de otimização combinatória que contém uma varável binária associada a cada variável contínua [Sun e McKeown, 1993]. Em 1968, Hirsch e Dantzig também mostraram que o PTCF é um problema do tipo *NP-Hard* [Hirsch e Dantzig, 1968], o que significa que ele possui ordem de complexidade exponencial.

O Problema de Transporte Linear por Partes (PTLP) surgiu como uma generalização natural do Problema de Transporte com Custo Fixo [Christensen e Labbe, 2015]. Nele, há uma estrutura linear por partes para representar o custo de envio de algum item entre uma origem e um destino. Assim, o custo é composto por uma parcela contínua, proporcional à quantidade transportada, e uma parcela fixa, associada à necessidade de utilização da rota. Logo, encontrar um fluxo de custo mínimo entre um conjunto de origens e destinos também é um problema *NP-Hard* [Sheng et al., 2006].

Devido à sua versatilidade, o PTLP surge em muitas aplicações práticas. Problemas de transporte que incluem descontos apresentam esse tipo de função de custo [Sheng et al., 2006]; e ela também pode ser utilizada para modelar diferentes formas de transporte [Croxton et al., 2003b], [Lapierre et al., 2004]. Além disso, problemas de planejamento e escala da produção, onde a quantidade produzida necessita ser dividida em porções menores ou um tempo de configuração precisa ser considerado, também podem ser modelados através de um problema de transporte com

essa característica na função custo [Sheng et al., 2006]. Outras aplicações também podem ser encontradas no domínio das telecomunicações e da logística.

Aplicações que trabalham com uma estrutura de rede, tal como o PTLP, pertencem a um conjunto que, segundo [Pardalos e Rosen, 1987] compõem os mais difíceis problemas dentro da otimização combinatória, denominados problemas de fluxo em redes não-convexos linear por partes.

3. Definição e formulação matemática

Seja $G(N, A)$ um grafo bipartido completo, que denota a direção sobre a qual um problema de transporte é definido, e para o qual $I = \{1, ..., m\}$ representa o conjunto de fornecedores (origens) e $J = \{1, ..., n\}$ o conjunto de consumidores (destinos). Nesse grafo, *N* denota o conjunto de nós de origem/destino, e *A* o conjunto de arcos (i, j) que conectam cada uma das origens *i* a cada um dos destinos *j*. A capacidade total de cada fornecedor *i* é representada por S_i, enquanto D_j representa a demanda total de cada consumidor *j*.

Cada arco (i, j) possui uma estrutura com $k_{i,j}$ segmentos de linha, também chamados de modos. Por conveniência de notação, assuma que $k_{i,j} = k$ para todo arco (i, j), e que $Q = \{1, ..., k\}$ denota o conjunto dos modos. Essa estrutura linear por partes representa o custo de cada unidade transportada da origem $i \in I$ para o destino $j \in J$. Cada modo $q \in Q$, no arco (i, j), é caracterizado por um custo fixo f_{ijq} para utilização da rota, e um custo variável c_{ijq} por unidade transportada.

Além disso, o fluxo usando o modo q no arco (i, j) é restrito a um mínimo $L_{ij,q-1}$ e um máximo L_{ijq}, onde $L_{ij0} = 0$. Considere que o fluxo máximo no arco (i, j) não pode exceder nem a capacidade S_i e nem a demanda D_j, isso é, $L_{ijk} \leq min\{S_i, D_j\}$.

Assim, o custo total de transporte no arco (i, j) é uma função linear por partes, separável, semi-contínua inferior, com k segmentos de linha definidos no intervalo de 0 a L_{ijk}, e esse custo aumenta a uma taxa decrescente, conforme o fluxo avança nos modos do arco. A representação do custo de transporte ao longo do arco (i, j) pode ser visualizada na Figura 1.

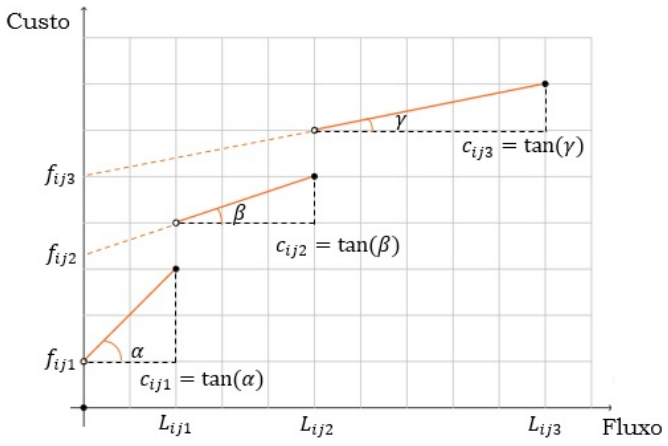

Figura 1: O custo como uma função linear por partes.

Um dos modelos de programação inteira-mista que pode ser utilizado para representar uma função descontínua e linear por partes é o modelo de múltipla escolha, [Balakrishnan e Graves, 1989], [Croxton et al., 2003a]. Adotando esse modelo, e considerando que a variável não-negativa x_{ijq} representa o fluxo entre *i* e *j* no modo *q*, e que a variável binária associada y_{ijq} assume valor 1 caso o modo seja usado e 0, em caso contrário, a formulação matemática do problema de transporte linear por partes pode ser descrita como:

$$Min \sum_{i \in I} \sum_{j \in J} \sum_{q \in Q} (c_{ijq} \cdot x_{ijq} + f_{ijq} \cdot y_{ijq}). \quad (1)$$

$$\sum_{i \in I} \sum_{q \in Q} x_{ijq} = D_j, \quad \forall j \in J. \quad (2)$$

$$\sum_{q \in Q} y_{ijq} \leq 1, \quad \forall (i,j), i \in I, j \in J. \quad (3)$$

$$\sum_{j \in J} \sum_{q \in Q} x_{ijq} \leq S_i, \quad \forall i \in I. \quad (4)$$

$$x_{ijq} \leq L_{ijq} \cdot yijq, \quad \forall (i,j,q), i \in I, j \in J, q \in Q. \quad (5)$$

$$x_{ijq} \geq L_{ij,q-1} \cdot yijq, \quad \forall (i,j,q), i \in I, j \in J, q \in Q. \quad (6)$$

$$x_{ijq} \geq 0, \quad \forall (i,j,q), i \in I, j \in J, q \in Q. \quad (7)$$

$$y_{ijq} \in \{0,1\}, \quad \forall (i,j,q), i \in I, j \in J, q \in Q. \quad (8)$$

A função objetivo (1) minimiza o custo total de abastecimento dos consumidores. Através de (2) cada consumidor deve receber uma quantidade igual a sua demanda. As restrições (3) garantem que um único modo seja usado para cada diferente combinação de origem e destino. As restrições (4) asseguram que a solução obedece às limitações de cada fornecedor, enquanto as restrições (5) e (6) restringem o fluxo dentro dos limites superior e inferior para o modo *q*, entre a origem *i* e o destino *j*.

4. Geração das instâncias e testes

Para a geração das instâncias de teste foram considerados, inicialmente, problemas com $n \in \{5, 10, 15\}$ fornecedores, $m \in \{15, 30, 50, 100\}$ consumidores e com $k \in \{3, 5\}$ modos, conforme considerado por [Christensen e Labbe, 2015]. Na sequência, também foram geradas mais seis instâncias de teste envolvendo apenas problemas quadrados de ordem 30, 50 e 100, também para $k \in \{3, 5\}$ modos.

A metodologia de geração das instâncias está baseada na proposta de [Christensen e Labbe, 2015], com adaptações, conforme segue. Considere que $U[x, y]$ denota a distribuição uniforme entre x e y e *UI* a versão discretizada dessa distribuição. A demanda do consumidor foi gerada a partir da distribuição $UI[20; 100]$ e a capacidade do fornecedor, a partir de $UI[0, 3 \cdot h; h]$, onde *h* denota a demanda esperada por fornecedor, nesse caso $h = (60/1.3) \cdot n/m$, de forma que o limite inferior da distribuição seja 30% do limite superior. O incremento na capacidade do arco foi gerado a partir de $UI[0, 8 \cdot l; 1, 2 \cdot l]$, onde *l* representa a demanda esperada por modo no arco em questão, isso é, $l = D_j/k, j \in J$ para cada um dos $(k-1)$ modos do arco $(i,j), i \in I, j \in J$. O incremento no último modo de cada arco corresponde ao valor necessário para atingir a demanda do arco.

O incremento no custo fixo foi gerado a partir da distribuição $UI[200, 1000]$, e custo variável do modo $(i,j,q), i \in I, j \in J, q \in Q$ calculado através de $U[0, 5 \cdot t; t]$, onde $t = (f_{i,j,q+1} - f_{ijq})/L_{ijq}, i \in I, j \in J, q \in Q$. Contudo, a partir do segundo modo de cada arco, o limite superior da distribuição passa a ser o custo variável do modo anterior, caso $t > c_{ij,q-1}, i \in I, j \in J, q \in Q$, de forma a garantir que o custo variável por unidade decresça conforme avançam os modos *q* dentro do arco $(i,j), i \in I, j \in J$.

Assim, há uma relação entre o custo fixo de utilização do modo e a sua capacidade. As instâncias têm custos descontínuos e não decrescentes, bem como capacidades e demandas inteiras. O custo é uma função linear por partes, separável, semi-contínua inferior, que aumenta a uma taxa decrescente, conforme o fluxo avança nos modos do arco.

Foram gerados 36 grupos de problemas, cada um com 30 instâncias, totalizando 1080 problemas. O desempenho dos *solvers* foi comparado, limitando o tempo de processamento (120 segundos), e analisado os resultados com base nos valores da função objetivo e do limitante, obtidos em cada problema considerado.

Para a geração das instâncias e posterior realização dos testes computacionais um programa, desenvolvido em linguagem Visual Basic 2015 a partir da plataforma do Visual Studio Community 2015 [Microsoft Corporation, 2015], foi criado. Esse programa permite que, por grupo, as instâncias de problemas sejam escolhidas e resolvidas pelos *solvers*, que também são chamados através do programa. Para a resolução dos problemas foram utilizados os *solvers* Gurobi Optimizer 7.0.1 [Gurobi Optimizer Inc., 2016] e IBM CPLEX Interactive 12.6.3 [IBM Corporation, 2015], uma vez que ambos são bem conceituados na área acadêmica e apresentam licenças bastante acessíveis.

Além disso, o programa também permite estipular um tempo limite de processamento para a resolução das instâncias de teste, gera o arquivo de solução para cada problema e, ao final, permite gerar também um arquivo com o resumo das informações de cada problema do lote, isso é, tempo de processamento, valor da função objetivo, do melhor limitante e do *gap*. Todos os problemas foram resolvidos utilizando uma mesma máquina com processador Core i7-4510U e 12 GB de memória RAM, e com ambos os *solvers* ajustados para a configuração *default*. O desempenho dos *solvers* foi analisado limitando o tempo de processamento em 120 segundos. Esse intervalo de tempo foi definido com base no objetivo da comparação, que é a futura aplicação no desenvolvimento de um método híbrido, para o qual pressupõe-se que o *solver* seja chamado diversas vezes para resolução do modelo. Assim, espera-se que, utilizando um tempo de processamento de 120 segundos, seja possível encontrar boas soluções viáveis sem, no entanto, gerar problemas de execução em função da memória computacional exigida. A comparação entre o CPLEX e o Gurobi foi realizada com base nos valores da função objetivo e do *gap*, obtidos em cada um dos problemas considerados. O percentual de melhoria de um *solver*, em comparação ao outro, foi verificado a partir do valor da função objetivo e do *gap* obtidos, de acordo com o grupo de problemas.

5. Resultados computacionais

Em 10 grupos de problemas, todas as instâncias, ou a maior parte delas, foram resolvidas na otimalidade antes de atingir o limite de tempo de processamento estabelecido (120 segundos). A Tabela 1 especifica quais são esses grupos e os resultados obtidos.

Tabela 1: Comparativo de tempo computacional entre os *solvers* para obtenção do ótimo.

Tamanho dos Problemas	**Número de Modos**	**Problemas que atingiram o ótimo**		
		No menor tempo		**Não atingiu o ótimo em 120s**
		Gurobi	**Cplex**	
5x15	3	10	20	-
	5	8	22	-
5x30	3	11	19	-
	5	6	24	-
5x50	3	22	8	-
	5	10	19	1
5x100	3	12	14	4
	5	13	14	3
10x15	3	-	24	6
	5	2	14	14

Dessa gama de grupos de problemas, a maior parte tem $m = 5$, que é o menor número de origens admitido. Apenas em dois grupos há 10 origens, contudo o número de destinos também é o menor admitido: apenas 15. Soluções ótimas não puderam ser obtidas para nenhum problema quadrado no limite de tempo estabelecido.

É possível observar que nesses casos, onde o *solver* atingiu o valor ótimo, o desempenho do Cplex foi melhor em 9 dos 10 grupos considerados. Do total de instâncias testadas para esses grupos, em 59,33% o Cplex apresentou vantagem, enquanto o Gurobi obteve vantagem em 31,33% dos problemas e em 9,33% deles ambos os *solvers* tiveram empate.

A Tabela 2 apresenta a relação dos grupos de problemas onde, na maior parte das instâncias, apenas soluções viáveis foram obtidas no tempo de processamento estabelecido. Nela, também estão especificados o número de problemas, por grupo, em que cada um dos *solvers* alcançou melhor desempenho em relação ao valor da função objetivo, bem como aqueles em que ambos empataram.

Tabela 2: Comparativo de melhor função objetivo alcançada pelos *solvers* com limite de tempo.

Tamanho dos Problemas	**Número de Modos**	**Problemas onde a função objetivo obteve**		
		Melhor resultado		**Empate**
		Gurobi	**Cplex**	
10x30	3	7	6	17
	5	13	3	14
10x50	3	12	3	15
	5	11	7	12
10x100	3	5	8	17
	5	9	6	15
15x15	3	3	6	21
	5	9	9	12
15x30	3	11	7	12
	5	12	8	10
15x50	3	12	3	15
	5	14	2	14
15x100	3	17	9	4
	5	15	6	9
30x30	3	13	15	2
	5	28	2	-
50x50	3	28	2	-
	5	28	2	-
100x100	3	28	2	-
	5	29	1	-

Os resultados mostram que, dos casos em que o *solver* não conseguiu alcançar o ótimo no tempo estabelecido, o desempenho do Gurobi foi melhor. Dos 20 grupos considerados, em 16 deles o número de problemas com melhor valor da função objetivo foi maior no Gurobi. Além disso, o Gurobi apresentou vantagem em 50,67% do total de instâncias testadas para esses grupos, enquanto o Cplex apresentou vantagem em apenas 17,83% e nos demais 31,5% os dois *solvers* empataram.

[Silva, 2012], estudando uma metodologia de resolução de problemas de transporte esparsos, observou que, tanto nos casos esparsos quando nos casos densos, o tempo de resolução de problemas quadrados é muito mais elevado quando comparado aos problemas retangulares. No estudo em questão, testes preliminares aplicados à alguns problemas quadrados, escolhidos aleatoriamente entre as instâncias geradas, mostraram que, mesmo em problemas de ordem 30, nem o Gurobi e nem o Cplex conseguiram encontrar a solução ótima no limite de tempo estabelecido, que foi de 24 horas.

Analisando apenas os problemas quadrados, é possível verificar que o ganho percentual que o Gurobi oferece em relação ao Cplex, no que diz respeito ao valor da função objetivo alcançado, cresce a medida que aumenta o tamanho dos problemas, apresentando um vantagem muito significativa (em torno de 47%) nos problemas de ordem 100. Essa vantagem pode ser observada através do gráfico da Figura 2.

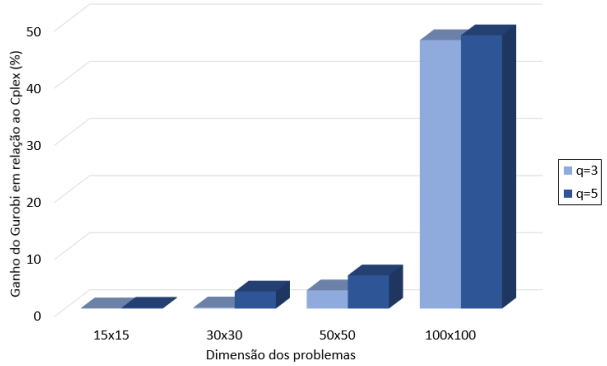

Figura 2: Melhoria percentual na função objetivo do Gurobi, conforme a variação no tamanho do problema.

O gráfico apresentado na Figura 3 mostra o ganho percentual proporcionado pelo Gurobi, quando comparado ao Cplex, no que tange ao *gap*, especificamente para os problemas quadrados.

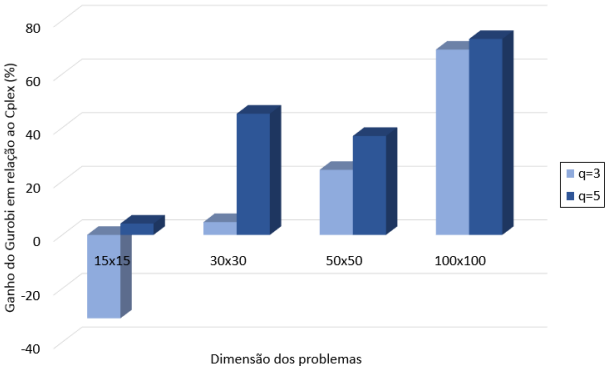

Figura 3: Melhoria percentual no *gap* do Gurobi, conforme a variação no tamanho do problema.

É possível verificar que, à medida que a ordem dos problemas aumenta, a vantagem do *solver* Gurobi também cresce, atingindo cerca de 71% para problemas de tamanho 100x100. Contudo, a análise do *gap* está diretamente associada ao valor do melhor limitante obtido e a sua razão com o valor da função objetivo alcançado. Dessa forma, sua análise precisa ser realizada observando conjuntamente os valores atingidos na função objetivo. Assim, a perda em relação ao *gap* que pode ser visualizada no gráfico, para os problemas de ordem 15 e $q = 3$, ocorre porque mesmo o Gurobi tendo apresentado pequeno ganho em relação à função objetivo, esse mesmo ganho não foi verificado para o valor do melhor limitante, fazendo com o *gap* apresentasse um valor negativo.

6. Considerações finais

Diante dos resultados encontrados, observa-se que o *solver* Gurobi apresentou melhor desempenho que o Cplex, considerando as condições impostas aos problemas e à resolução. Apesar do Cplex mostrar-se vantajoso na resolução de problemas pequenos, para os quais atingiu o ótimo em menor tempo, a medida que o tamanho dos problemas cresce, o desempenho do Gurobi, nas instâncias testadas, tende a ser melhor, considerando o valor da função objetivo encontrada e o limite de tempo de processamento estabelecido. É possível destacar ainda o crescente ganho percentual do Gurobi sobre o Cplex, no que tange ao valor da função objetivo e ao *gap*, quando avaliados os problemas quadrados, os quais apresentam maior grau de dificuldade de resolução.

Assim, no processo de escolha de um *solver* para aplicação no desenvolvimento de um método híbrido de resolução do PTLP, o Gurobi parece ser uma escolha mais vantajosa.

7. Agradecimento

Os autores agradecem o apoio financeiro recebido da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e dos Institutos Federais de Educação, Ciência e Tecnologia de Santa Catarina (IFSC) e do Paraná (IFPR).

Referências

Balakrishnan, A. e Graves, S. C. (1989). A composite algorithm for a concave-cost network flow problem. *Networks*, 19:175–202.

Christensen, T. R. L. e Labbe, M. (2015). A branch-ct-and-price algorithm for the piecewise linear trasnportation problem. *European Journal of Operational Research*, 245:645–655.

Croxton, K. L., Gendron, B., e Magnanti, T. L. (2003a). A comparasion of mixed-integer programming models for nonconvex piecewise linear cost minimization problem. *Management Science*, 49:1268–1273.

Croxton, K. L., Gendron, B., e Magnanti, T. L. (2003b). Models and methods for merge-in-transit operations. *Transportation Science*, 37:1–22.

Gurobi Optimizer Inc. (2016). Gurobi optimizer 7.0.1.

Hirsch, W. M. e Dantzig, G. B. (1954). *Notes on linear programming: Part XIX - The fixed charge problem. Memorandum 1383*. RAND Research, Santa Monica.

Hirsch, W. M. e Dantzig, G. B. (1968). The fixed charge problem. *Naval Research Logistics Quarterly*, 15:413–424.

IBM Corporation (2015). IBM ILOG CPLEX Interactive Optimizer 12.6.3.0.

Lapierre, S., Ruiz, A. B., e Soriano, P. (2004). Designing distribuiton networks: formulation and solution heuristic. *Transportation Science*, 38:174–187.

Microsoft Corporation (2015). Microsoft visual studio community.

Nemhauser, G. L. e Wolsey, L. A. (1988). *Integer and Combinatorial Optimization*. John Wiley and Sons Inc., New York.

Pardalos, P. M. e Rosen, J. B. (1987). *Constrained Global Optimization: Algorithms and Applications*. Springer-Verlag, Berlin.

Sheng, S., Dechen, Z., e Xiofei, X. (2006). Genetic algorithm for the transportation problem with discontinuos piecewise linear cost function. *International Journal of Computer Science and Network Security*, 6:182–189.

Silva, T. C. L. (2012). *Nova Metodologia para Resolucao de Problemas de Transporte em Casos Esparsos*. PhD thesis, Universidade Federal do Parana.

Sun, M. e McKeown, P. G. (1993). Tabu search applied to the general fixed charge problem. *Annals of Operational Research*, 41:405–420.