Problema de estoque e roteirização com empacotamento bidimensional

Pedro Henrique Del Bianco Hokama
Universidade Federal de São Carlos
hokama@ic.unicamp.br

Flávio Keidi Miyazawa
Universidade Estadual de Campinas
fkm@ic.unicamp.br

Reinaldo Morabito Neto
Universidade Federal de São Carlos
morabito@ufscar.br

RESUMO

Neste trabalho investigamos o problema de estoque e roteirização com empacotamento bidimensional, no qual precisamos atender pedidos de diversos clientes em um horizonte de planejamento. Os pedidos contêm múltiplos produtos distintos e, para as entregas, são utilizados múltiplos veículos de uma frota homogênea. Tanto os produtos como os contêineres dos veículos são retângulos bidimensionais. O problema pode ser diretamente aplicado no transporte de bens e ainda não foi investigado na literatura.

Trata-se de um problema NP-Difícil, que envolve ainda encontrar empacotamentos para cada uma das rotas executadas pelos veículos, sendo esse também um problema NP-Difícil. Propomos um algoritmo *branch-and-cut*, que utiliza programação linear inteira e programação por restrições para encontrar boas soluções para o problema. Testes computacionais foram executados em instâncias de pequeno e médio porte e mostram que o modelo híbrido proposto é capaz de gerar soluções em tempos razoáveis.

PALAVRAS CHAVE. Problema de estoque e roteirização, Problema de empacotamento, Algoritmo híbridos.

ABSTRACT

In this paper we investigate the inventory-routing problem with two-dimensional loading, in which it is required to attend orders from several clients in a planning horizon. The orders contains multiple products, and the deliveries are performed by multiple vehicles from a homogeneous fleet. Products and the containers attached to the vehicle are two-dimensional rectangles. The problem is highly applicable in the transport of goods and has not yet being investigated in the literature.

This is a NP-Hard problem, that requires to find a packing for each one of the routes performed by the vehicles, which is also a NP-Hard problem. We propose a branch-and-cut algorithm, using integer linear programming and constraint programming to find solutions to the problem. Computational tests were performed in small and medium sized instances and show that the proposed hybrid model is a able to find solutions in reasonable computational time.

KEYWORDS. Inventory-Routing Problem. Conteiner Loading Problem. Hybrid Models.

Introdução

Estamos investigando métodos para encontrar boas soluções para o problema de estoque e roteirização com empacotamento bidimensional (*Inventory-Routing Problem with Two-dimensional Loading* - IRP2D). Nesse problema um fornecedor precisa atender as demandas de seus clientes (ou de seus próprios pontos de vendas) em um determinado horizonte de planejamento utilizando um ou mais veículo. Tanto os produtos como os contêineres são considerados em formato bidimensional retangular, caso comum no empacotamento de itens grandes ou frágeis, ou ainda de produtos transportados em paletes. O objetivo do problema é minimizar os custos de deslocamento dos veículos e, para cada rota, encontrar um empacotamento dos itens dentro do contêiner. O empacotamento deve permitir que em cada visita os itens entregues possam ser retirados em um único movimento sem a necessidade de se remanejar os itens restantes.

O problema de estoque e roteirização na sua versão unidimensional (*Inventory-Routing Problem* - IRP). foi primeiramente apresentado por [Bell et al., 1983], desde então o problema foi investigado com várias variantes. Recomendamos o trabalho de [Coelho et al., 2013] em homenagem aos trinta anos do problema, onde os autores apresentam uma revisão bibliográfica do IRP e suas variantes. No trabalho de [Coelho e Laporte, 2013] os autores apresentam algoritmos exatos para a resolução de diversas variantes do problema. Já no trabalho de [Archetti et al., 2007], heurísticas são apresentadas para o IRP.

Problemas que envolvem roteamento e empacotamento já foram estudados na literatura. Para o problema de roteamento de veículos com empacotamento bidimensional um dos primeiros e principais trabalhos é apresentado por [Iori et al., 2007]. Mais recentemente o problema foi tratado por [Hokama et al., 2016] que apresenta um algoritmo *branch-and-cut* utilizando programação linear inteira e programação por restrições. Em ambos os trabalhos no sub-problema de empacotamento bidimensional, rotações dos itens não são consideradas.

Dentro da tipologia apresentada por Coelho, estamos inicialmente interessados no problema de estoque e roteirização restrito, com horizonte de planejamento finito, de um fornecedor para muitos clientes, com roteamento múltiplo (onde cada rota pode atender mais de um cliente), entrega flexível (onde a quantidade de produtos entregue não é fixa) frota homogênea com múltiplos veículos, além da restrição de empacotamento. A figura 1 apresenta um exemplo de instância com três clientes e dois períodos no horizonte de planejamento, os itens precisam ser entregues até o período em que são apresentados.

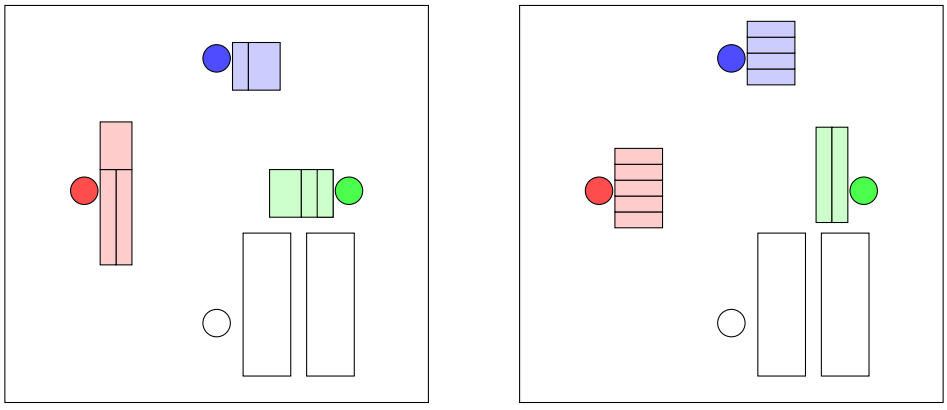

Figura 1: Exemplo de uma instância com três clientes e horizonte de planejamento de tamanho 2, os círculos coloridos representam os clientes e próximo a eles os itens dos pedidos em cada período. O círculo branco representa o depósito e próximo a ele os veículos disponíveis.

Por se tratar de um problema novo, não existem resultados ou instâncias na literatura para esse problema, por isso comparamos as soluções obtidas apenas pela distância da relaxação linear.

As instâncias foram geradas aleatoriamente baseadas nas instâncias para o problema roteamento de veículos.

Na próxima sessão explicamos o problema formalmente seguido pela formulação em programação linear inteira que estamos considerando para o problema. Posteriormente descrevemos o funcionamento do algoritmo de *branch-and-cut* proposto, incluindo o modelo de programação por restrições (*Constraint Programming* - CP) que utilizamos para resolver o problema de empacotamento bidimensional. Por fim apresentamos os resultados computacionais obtidos com esse algoritmo, as conclusões e possíveis desdobramentos dessa pesquisa.

Definição do problema

O problema de estoque e roteirização com empacotamento bidimensional (IRP2D), é bastante relevante para a distribuição e armazenamento de produtos que possuem alguma forma definida. Contrastante com a versão clássica do problema que considera apenas uma dimensão dos produtos, o que em geral é suficiente para transporte de gases, líquidos e objetos em que o peso é mais restritivo que a forma deles, mas não é suficiente no transporte de produtos como caixas e paletes.

Na versão do problema que estamos interessados, consideramos que existem diferentes produtos, com diferentes dimensões, que precisam ser entregues (múltiplos produtos). Para realizar as estregas está disponível um certo número de veículos que podemos usar (múltiplos veículos), estes são padronizados em possuem as mesmas dimensões (frota homogênea).

Os clientes possuem pedidos (notas fiscais) que são compostos por diversos itens que devem ser entregues até um determinado período de tempo. Prática comum no transporte de cargas, esse pedido é indivisível e deve ser entregue em um único caminhão (*no-split*). A razão disso é que geralmente um pedido deve ser fechado em uma única nota fiscal que pelas leis brasileiras devem transitar junto aos produtos nela contidos. Formalmente o IRP2D recebe como entrada:

- Um grafo não direcionado completo, $G = (V, E)$, onde $V = \{0, \ldots, |V|\}$.
- Um fornecedor representado pelo vértice $0 \in V$.
- Uma função de custo para as arestas do grafo, $c : E \to \mathbb{Q}$.
- Um conjunto de vértices $V' = V \backslash \{0\}$ representando os clientes.
- Uma lista de produtos $B = \{0, \ldots, |B|\}$ oferecida pelo fornecedor.
- As dimensões (l_b^1, l_b^2) do item $b \in B$, que pode ser rotacionado.
- Uma área máxima A_v, em cada cliente $v \in V'$ para estoque dos produtos.
- Um horizonte de planejamento T.
- Uma demanda $d_{v,t}^b$ em cada cliente $v \in V'$ de quantos itens do tipo $b \in B$ o cliente deseja receber até o período t.
- Um conjunto de K veículos com um contêiner de dimensões (W, H), com uma porta em um dos lados.

O objetivo é encontrar uma solução de custo mínimo que deve atender as seguintes características:

1. O fornecedor é um grande depósito e para todos os efeitos os estoques de produtos nele podem ser considerados ilimitados.
2. Cada veículo é capaz de atender uma rota por período, saindo e retornando ao fornecedor, atendendo à um subconjunto de clientes.
3. O estoque de cada cliente não pode ultrapassar a área de seu depósito.
4. As demandas de um cliente em um dado período são consideradas um único pedido, e devem ser entregues em um único caminhão.

5. Cada cliente só pode receber a visita de no máximo um caminhão por período, contendo um ou mais pedidos.
6. Os itens de uma rota devem ser empacotados no contêiner do veículo, podendo ser rotacionados 90 graus.
7. As dimensões do contêiner não podem ser ultrapassadas.
8. Os itens não podem se sobrepor.
9. Em cada visita os itens devem ser descarregados com um único movimento em direção a porta do contêiner (restrição de descarregamento).

A Figura 2 apresenta uma solução ótima para a instância do problema apresentado na Figura 1.

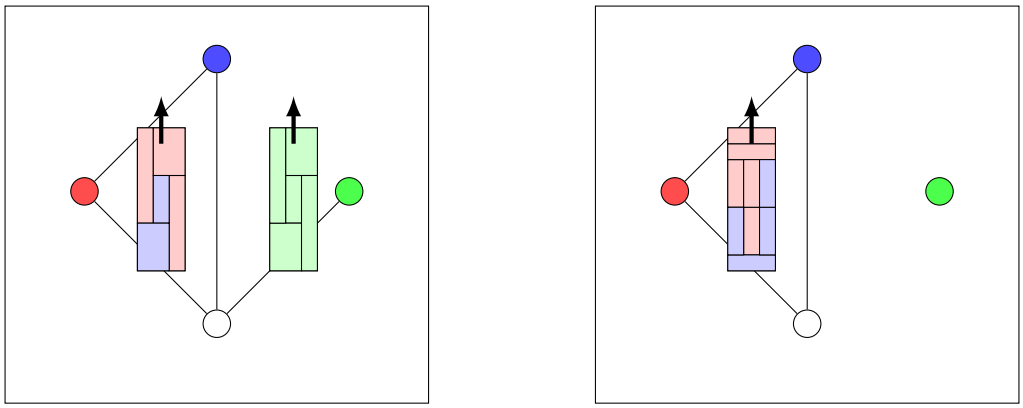

Figura 2: Exemplo de solução ótima para a instância apresentada na Figura 1. No primeiro período são utilizados dois veículos, um contendo os pedidos do primeiro período para o cliente azul (mais a esquerda) e vermelho (mais acima), o outro veículo entrega todos os pedidos do cliente verde (mais a direita). No segundo período apenas um veículo é necessário para entregar os pedidos restantes.

Formulação

Para formular o problema vamos considerar que as demandas $d^t_{v,b}$ de um mesmo cliente em um mesmo período são dadas por um único pedido ϕ. O pedido ϕ deve ser entregue ao cliente $v(\phi)$ no período $t(\phi)$ ou em qualquer período anterior, desde que o conjunto de produtos já estocados no cliente não exceda o limite permitido. Para simplificar a notação, considere também que $d^b_\phi = d^b_{v(\phi),t(\phi)}$. O conjunto de todos os pedidos é representado pelo símbolo Φ.

Considere $\Phi(t) = \{\phi \in \Phi | t(\phi) \leq t\}$, o conjunto de pedidos que podem ser atendidos no período t e $\Phi(v,t) = \{\phi \in \Phi | v(\phi) = v \text{ e } t(\phi) \leq t\}$, o conjunto de pedidos do cliente v que podem ser atendidos no período t. Uma determinada sequência de clientes com determinados pedidos associados a cada um deles, é chamado de rota. Uma rota é dita inviável se não existe um empacotamento dos respectivos produtos que respeite as restrições consideradas. O conjunto de todas as rotas inviáveis é definido como $\mathcal{R}$.

Iremos considerar as seguintes variáveis: Seja x^{tk}_e uma variável que indica o número de vezes que a aresta e é percorrida pelo veículo k no período t. A variável binária y^{tk}_0 assume valor 1 se o veículo k executa uma rota no período t e 0 caso contrário. Enquanto a variável y^{tk}_v assume valor 1 se o veículo k visita o cliente v no período t e 0 caso contrário. Seja q^{tk}_ϕ uma variável binária que assume valor 1 se o veículo k atende o pedido ϕ no período t.

$$\min \quad \sum_{e\in E}\sum_{k\in K}\sum_{t\in T} c_e x_e^{tk}, \tag{1}$$

subject to

$$\sum_{b\in B}\sum_{\phi\in\Phi(t)} (d_\phi^b\, l_b^1\, l_b^2) q_\phi^{kt} \le W\, H\, y_0^{kt}, \quad k\in K, t\in T, \tag{2}$$

$$\sum_{t\le t(\phi)}\sum_{k\in K} q_\phi^{tk} = 1, \quad \phi\in\Phi \tag{3}$$

$$q_\phi^{tk} \le y_{v(\phi)}^{tk}, \quad \phi\in\Phi, k\in K, t\in T, \tag{4}$$

$$\sum_{\phi\in\Phi(v,t)}\sum_{k\in K}\sum_{t'\le t}\sum_{b\in B} (d_\phi^b l_b^1\, l_b^2) q_\phi^{kt'} \le A_v, \quad v\in V', t\in T, \tag{5}$$

$$\sum_{e\in\delta(i)} x_e^{tk} = 2y_i^{tk}, \quad i\in V, k\in K, t\in T, \tag{6}$$

$$\sum_{e\in\delta'(\mathcal{V})} x_e^{tk} \le \sum_{i\in\mathcal{V}} y_i^{tk} - y_m^{tk}, \quad \mathcal{V}\subset V, t\in T, k\in K, m\in\mathcal{V}, \tag{7}$$

$$\sum_{e\in R} x_e^{tk} + \sum_{\phi\in D} q_\phi^{tk} \le |R| + |D| - 1, \quad (R,D)\in\mathcal{R}, \tag{8}$$

$$x_e^{tk}\in\{0,1\}, \quad e\in E\backslash\delta(0), k\in K, t\in T, \tag{9}$$

$$x_e^{tk}\in\{0,1,2\}, \quad e\in\delta(0), t\in T, \tag{10}$$

$$y_i^{tk}\in\{0,1\}, \quad i\in V, k\in K, t\in T, \tag{11}$$

$$q_\phi^{tk}\in\{0,1\}, \quad \phi\in\Phi, k\in K, t\in T. \tag{12}$$

As desigualdades (2) garantem que em cada período t, a soma das áreas dos itens dos pedidos atendidas pelo veículo K, não ultrapasse a área do contêiner, como veremos, essa desigualdade é redundante com a (8), mas é mantida por questão de performance.

As desigualdades (3) garantem que todo pedido será atendido exatamente uma vez, por um único veículo. As desigualdades (4) garante que um pedido de um cliente só pode ser atendido por um veículo que o visita. As desigualdades (5) garantem que a área utilizadas por todos os pedidos já atendidos mas em estoque no cliente não ultrapasse a área máxima disponível. As desigualdades de (6) e (7) garantem o grau e a conectividade das rotas. As desigualdades (8) garantem que a solução não contenha nenhuma rota inviável. As demais linhas garantem a correta atribuição das variáveis.

Algoritmo branch-and-cut para o IRP2D

Na formulação apresentada, as linhas (7) e (8) apresentam cada uma um número exponencial de desigualdades, por essa razão, para qualquer instância não trivial é inviável inserir todas essas desigualdades no modelo. Propomos então um algoritmo *branch-and-cut* onde essas desigualdades não são inseridas inicialmente. Ao invés disso resolvemos o modelo sem essas restrições, que chamamos de modelo relaxado, utilizando um resolvedor de programação linear. O resolvedor utiliza um algoritmo *branch-and-bound* para encontrar soluções inteiras para o modelo relaxado, chamaremos essas de *soluções parciais*. Sempre que o resolvedor encontra uma solução parcial, verificamos se ela viola alguma das desigualdades da linha (7) ou (8). Caso uma violação seja detectada, inserimos a desigualdade violada como uma *lazy constraint*, que eliminará a solução infactível. Então o modelo é re-otimizado.

Uma *lazy constraint* é uma desigualdade que necessariamente deve ser satisfeita para a corretude de uma solução, porém como raramente ela é violada podemos deixar para adiciona-la ao modelo somente quando for necessária. No nosso caso de todas as desigualdades (uma quantidade exponencial) apenas algumas são violadas durante a resolução do modelo. A solução final devolvida pelo resolvedor não pode violar nenhuma das desigualdades. Para verificar se uma solução parcial viola uma dessas desigualdades precisamos de um algoritmo que as verifique.

Dois tipos de violações acontecem no modelo relaxado. A primeira é a presença de sub-ciclos, que não passam pelo depósito. Esse caso é conhecido na literatura por ocorrer na maioria dos modelos descritos para o problema do caixeiro viajante. Detectado um sub-ciclos que passa pelos vértices $\mathcal{V} \in V$ mas não passa pelo depósito, realizado pelo veículo k no período t, adicionamos para cada $m \in \mathcal{V}$ a desigualdade:

$$\sum_{e \in \delta'(\mathcal{V})} x_e^{tk} \leq \sum_{i \in \mathcal{V}} y_i^{tk} - y_m^{tk}, \quad \mathcal{V} \subset V, t \in T, k \in K,$$

Essa desigualdade também foi utilizada nos trabalhos de [Coelho e Laporte, 2013]

Quando nenhum sub-ciclo é detectado, ainda é necessário verificar cada uma das rotas daquela solução parcial possui um empacotamento viável. Descobrir se um determinado conjunto de itens pode ser empacotado em um contêiner também é um problema NP-Difícil, e por isso não se conhece um algoritmo de tempo polinomial que resolva o problema de forma exata. Nesse trabalho estamos propondo uma modelagem em Programação por Restrições (*Constraint Programming* - CP) para resolver esse problema. O modelo deve permitir a rotação em 90 graus dos itens, bem como garantir que os itens podem ser descarregados em um único movimento na direção da porta do contêiner.

Modelo em CP para o problema do empacotamento: Seja P o conjunto de coordenadas de discretização no eixo X, e Q o conjunto de coordenadas de discretização no eixo Y. Seja ainda $D_i = l_i^1, l_i^2$ as dimensões do item i.

Considere as variáveis inteiras x_i e y_i que indicam a posição (x_i, y_i) onde o item i será posicionado. E as variáveis inteiras w_i, h_i que indicam as dimensões do item i paralelas ao eixo X e ao eixo Y respectivamente, de acordo com a rotação do item. Note que as variáveis x_i e y_i aqui não tem relação com as variáveis de roteamento. Atribuímos aos domínios das variáveis:

$$x_i \in P, \qquad y_i \in Q, \qquad w_i \in D_i, \qquad h_i \in D_i. \tag{13}$$

As restrições necessárias para garantir que os itens não ultrapassam o limite do contêiner são dadas por:

$$x_i + w_i \leq W, \qquad y_i + h_i \leq H. \tag{14}$$

Além disso para garantir a não sobreposição de itens, para cada par de itens i, j que pertencem ao mesmo cliente, ou seja, que a ordem de descarregamento entre eles é indiferente, adicionamos a seguinte restrição de não sobreposição:

$$x_i + w_i \leq x_j \quad \text{ou} \quad y_i + h_i \leq y_j \quad \text{ou} \quad y_j + h_j \leq y_i \quad \text{ou} \quad x_j + w_j \leq x_i. \tag{15}$$

No caso de itens de diferentes clientes, considerando que o item i deve ser descarregado antes do item j, adicionamos a seguinte restrição:

$$x_i + w_i \leq x_j \quad \text{ou} \quad y_j + h_j \leq y_i \quad \text{ou} \quad x_j + w_j \leq x_i. \tag{16}$$

Além disso como permitimos rotações, ou seja, w_i e h_i são intercambiáveis, se $l_i^1 \neq l_i^2$ então adicionamos:

$$w_i \neq h_i. \tag{17}$$

Resultados Computacionais

Todo o código foi implementado em linguagem c++, compilado e executado em ambiente linux. O resolvedor de programação linear inteira utilizado foi o Gurobi 7.0.2, o resolvedor de programação por restrições foi o CP Optimizer 12.5. Todo o código e intâncias estão disponíveis em http://www.hokama.com.br/irp2d_sbpo/.

Foram geradas instâncias considerando 5, 10 e 15 clientes. Os itens foram gerados de maneira semelhante aquela utilizada por [Iori et al., 2007] para o problema de roteamento. Quatro classes de instâncias foram geradas, são elas $r \in \{2, 3, 4, 5\}$ Em cada em cada período, o número de itens que cada cliente demanda é uniformemente sorteado em $[1, r]$.

O número de diferentes tipos de itens distintos B variando em $B = \{5, 10, 15\}$. O número de veículos disponíveis é considerado aquele necessário para atender as demandas se apenas 50% da área útil do caminhão for utilizada. O número de períodos considerados foi de 3, 5 e 7. As dimensões do veículo são $W = 20$ e $H = 40$. E as dimensões dos itens seguem o trabalho de [Iori et al., 2007]

As tabelas 1, 2 e 3 apresentam os resultados obtidos para as instâncias com três, cinco e sete períodos respectivamente. O tempo limite para a resolução foi de 600 segundos, sendo que para verificar se uma rota possui um empacotamento viável, são permitidos ao resolvedor de programação por restrições 10 segundos.

Tabela 1: Resultados para um horizonte de planejamento de tamanho 3

		V' = 5		V' = 10		V' = 15	
		gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)
Class 2	B = 5	0,00	1,12	0,00	224,3	41,92	-
	B = 10	0,00	0,86	0,00	107,08	36,47	-
	B = 15	0,00	0,37	0,81	-	39,91	-
Class 3	B = 5	0,00	1,58	19,31	-	48,23	-
	B = 10	0,00	0,82	17,53	-	42,24	-
	B = 15	0,00	1,16	21,44	-	45,86	-
Class 4	B = 5	0,00	1,31	0,00	114,18	29,70	-
	B = 10	0,00	1,2	9,68	-	42,85	-
	B = 15	0,00	0,91	16,88	-	35,50	-
Class 5	B = 5	0,00	1,88	0,00	25,37	15,58	-
	B = 10	0,00	0,91	0,00	21,32	11,82	-
	B = 15	0,00	2,5	0,00	38,12	16,73	-

Conclusões

Propomos um algoritmo *branch-and-cut* para o problema de estoque e roteirização com empacotamento bidimensional, e para o nosso conhecimento esse é o primeiro trabalho a abordá-lo. Os resultados computacionais indicam que essa é uma abordagem viável para o problema, sendo capaz de obter soluções, ainda que com um gap, para todas as instâncias testadas. O modelo apresentado pode contemplar diversas melhorias, como por exemplo, quebra de simetrias, cortes utilizados em problemas de roteamento, restrições redundantes para o problema de empacotamento. Além disso o modelo permite agregar, com certa clareza, outras restrições comuns na prática, como janelas de tempo, restrições de capacidade (peso) dos veículos, restrições de balanceamento de carga no empacotamento etc.

Referências

Archetti, C., Bertazzi, L., Laporte, G., e Speranza, M. G. (2007). A branch-and-cut algorithm for a vendor-managed inventory-routing problem. *Transportation Science*, 41(3):382–391.

Tabela 2: Resultados para um horizonte de planejamento de tamanho 5

		V' = 5		V' = 10		V' = 15	
		gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)
Class 2	B = 5	0,00	2,63	23,65	-	55,59	-
	B = 10	0,00	173,23	38,76	-	58,54	-
	B = 15	0,00	79,8	27,73	-	60,70	-
Class 3	B = 5	0,00	178,77	23,21	-	49,41	-
	B = 10	0,00	12,99	28,15	-	49,69	-
	B = 15	0,00	243,56	26,41	-	45,98	-
Class 4	B = 5	0,00	2,15	27,49	-	59,77	-
	B = 10	0,00	7,39	33,73	-	61,70	-
	B = 15	0,00	72,52	34,08	-	61,21	-
Class 5	B = 5	0,00	4,98	7,55	-	46,79	-
	B = 10	0,00	4,37	10,32	-	43,25	-
	B = 15	0,00	13,1	10,15	-	38,55	-

Tabela 3: Resultados para um horizonte de planejamento de tamanho 7

		V' = 5		V' = 10		V' = 15	
		gap(%)	time(s)	gap(%)	time(s)	gap(%)	time(s)
Class 2	B = 5	7,58	-	26,34	-	63,63	-
	B = 10	10,34	-	29,56	-	56,97	-
	B = 15	7,81	-	24,63	-	63,78	-
Class 3	B = 5	0,00	101,67	54,13	-	73,87	-
	B = 10	11,60	-	61,43	-	71,95	-
	B = 15	12,89	-	64,24	-	74,96	-
Class 4	B = 5	9,19	-	53,84	-	59,13	-
	B = 10	0,00	294,58	55,87	-	66,42	-
	B = 15	19,06	-	47,94	-	55,20	-
Class 5	B = 5	0,00	87,59	27,27	-	50,70	-
	B = 10	0,00	62,2	23,90	-	43,46	-
	B = 15	0,00	203,12	25,28	-	46,38	-

Bell, W. J., Dalberto, L. M., Fisher, M. L., Greenfield, A. J., Jaikumar, R., Kedia, P., Mack, R. G., e Prutzman, P. J. (1983). Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer. *Interfaces*, 13(6):4–23.

Coelho, L. C., Cordeau, J.-F., e Laporte, G. (2013). Thirty years of inventory routing. *Transportation Science*, 48(1):1–19.

Coelho, L. C. e Laporte, G. (2013). The exact solution of several classes of inventory-routing problems. *Computers & Operations Research*, 40(2):558 – 565. ISSN 0305-0548. URL http://www.sciencedirect.com/science/article/pii/S0305054812001773.

Hokama, P., Miyazawa, F. K., e Xavier, E. C. (2016). A branch-and-cut approach for the vehicle routing problem with loading constraints. *Expert Systems with Applications*, 47:1 – 13. ISSN 0957-4174. URL http://www.sciencedirect.com/science/article/pii/S0957417415007009.

Iori, M., Salazar-González, J.-J., e Vigo, D. (2007). An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science*, 41(2):253–264. URL `http://pubsonline.informs.org/doi/abs/10.1287/trsc.1060.0165`.