UM ALGORITMO *BRANCH AND CUT* PARA O PROBLEMA DE ROTEIRIZAÇÃO E ESTOQUES COM MÚLTIPLOS DEPÓSITOS E ENTREGAS FRACIONADAS

Cleder Marcos Schenekemberg
GTAO - Grupo de Tecnologia Aplicada à Otimização/UFPR
cledercms@hotmail.com

Thiago André Guimarães
GTAO - Grupo de Tecnologia Aplicada à Otimização/UFPR
thiagoandre@ufpr.br

Cassius Tadeu Scarpin
GTAO - Grupo de Tecnologia Aplicada à Otimização/UFPR
cassiusts@gmail.com

RESUMO

Neste trabalho resolvemos um Problema de Roteirização e Estoques com Múltiplos Depósitos e Entregas Fracionadas (Multi Depot Inventory Routing Problem with Split Deliveries - MDIRPSD). O Problema emerge no paradigma do sistemas de estoque gerenciado pelo fornecedor, quando as entregas são realizadas a partir de múltiplos depósitos e os clientes podem ser atendidos múltiplas vezes em um mesmo período. O MDIRPSD incorpora as decisões de suprimento e distribuição de forma simultânea e foi recentemente proposto na literatura. Nós apresentamos uma nova formulação matemática, propomos uma nova política de estoque e incorporamos os custos de estocagem dos clientes na otimização do problema. Também projetamos e implementamos um método exato que obteve resultados superiores ao branch-and-cut existente na literatura. Por fim, reportamos resultados comparativos entre as duas políticas de estoque consideradas.

Palavra-chave: Estoque Gerenciado pelo Fornecedor; Múltiplos Depósitos; Entregas Fracionadas; Método Exato.

ABSTRACT

In this paper we solve a Multi Depot Inventory Routing Problem with Split Deliveries (MDIRPSD). The problem emerges under the vendor-managed inventory systems paradigm, when deliveries are made from multiple depots and the clients can be served several times over the same period. The MDIRPSD integrates supply and distribution decisions and has recently been proposed in the literature. We present a new mathematical formulation, taking into account inventory holding costs and also propose a new inventory policy. We have also designed and implemented a new branch-and-cut that overcome the similar approach in the literature. Finally we report comparative results between the two inventory policies considered.

Keywords: Vendor-managed inventory; Multi-depot; Split deliveries; Branch-and-cut.

1. INTRODUÇÃO

Nas últimas décadas, a escalada competitiva global e a consequente redução das margens de lucro vem ensejando maior coordenação dos processos e atores no âmbito da cadeia de suprimentos. Ao mesmo tempo, o advento da indústria 4.0 e a disponibilização dados em tempo real, possibilitam uma agenda cada vez mais colaborativa nas atividades logísticas, com especial ênfase no elo fornecedor-cliente. Neste sentido, paradigmas que requerem a troca de informações e que esbarravam nos obstáculos tecnológicos de anos atrás, vem assumindo um protagonismo cada vez mais transversal. Não apenas pela redução nos custos dos sistemas de troca eletrônica de dados, mas também na robusta minimização de erros e incertezas que esses sistemas possibilitam, em comparação com técnicas indiretas de quantificação das demandas.

Os sistemas *Vendor Managed Inventory* (*VMI*) ou Estoque Gerenciado pelo Fornecedor, configuram uma prática gerencial colaborativa entre fornecedor e cliente, com inúmeros relatos de sucesso em diferentes segmentos (ver [1]). Sob um sistema *VMI*, o fornecedor controla o estoque do cliente, decidindo quando atendê-lo e quanto entregar por ocasião de um atendimento. A dinâmica dos estoques é diretamente dependente da demanda do cliente, que pode ser estimada por métodos de previsão, ou observada de forma precisa, através dos supracitados sistemas de troca eletrônica de dados. Segundo [2], a prática *VMI* é mutuamente benéfica, pois os clientes não precisam dispender recursos para controlar seus estoques e emitir pedidos de ressuprimento, enquanto os fornecedores ganham pela maior coordenação das atividades logísticas, particularmente na composição dos roteiros de entrega.

Pelo ângulo operacional, a implementação de um sistema *VMI* implica na resolução de um problema de otimização combinatorial complexo, denominado *Inventory-Routing Problem* (*IRP*) ou Problema de Roteirização e Estoques. O *IRP* integra, em um mesmo arcabouço, o problema de gerenciamento de estoques e o problema de roteamento de veículos com múltiplos períodos. Nesta abordagem conjunta, compete ao fornecedor decidir: quando servir um cliente, quanto entregar a cada visita realizada e como definir as rotas de entrega. Tais decisões devem garantir que o cliente tenha sua demanda plenamente atendida, seja pelos estoques mantidos e/ou pelas quantidades recebidas.

Embora a literatura sobre o *IRP* reporte diversas variantes (ver [3]), a estrutura logística mais comum é bastante simplificada, envolvendo apenas um fornecedor, que opera a partir de um único depósito, e múltiplos clientes, com a proibição de entregas fracionadas [4]. Todavia, em um contexto mais realístico, os clientes estão dispersos em uma ampla área urbana e com elevada densidade de tráfego. Ademais, o fornecedor geralmente dispõe de uma rede de depósitos e uma frota de veículos, o que eleva consideravelmente o número de opções distintas de servir a um cliente. Por esses aspectos, a possibilidade de entregas fracionadas se adere ao melhor uso dos recursos físicos do fornecedor (frota de veículos e múltiplos depósitos), ao mesmo tempo em que gerencia com mais eficiência as questões de tráfego e de logística urbana. Essas características definem um *IRP* com múltiplos depósitos e entregas fracionadas, a qual denominamos de *Multi Depot Inventory Routing Problem with Split Deliveries* (*MDIRPSD*), e que recentemente foi proposta por [5]. No trabalho, os autores introduzem o *MDIRPSD* e desenvolvem um algoritmo exato do tipo *branch-and-cut*, além de uma matheurística para resolver instâncias de maior porte. Ademais, o problema abordado não contempla o custo de estoque nos clientes, prática muito comum no *IRP*, (ver [6, 7, 8, 4]), e os autores consideram apenas a política de estoque *Maximum Level* (*ML*), quando o fornecedor está livre para quantificar a entrega ao cliente, limitada à capacidade de estoque no período.

Neste artigo, ampliamos o estudo do *MDIRPSD* e reformulamos o problema, a fim de incorporar o custo de estoque nos clientes nas decisões do fornecedor. Outrossim, introduzimos a política de estoque *Order-up to Level* (OU) no âmbito do *MDIRPSD*. A política OU foi proposta por [9] para o *IRP* clássico e é frequentemente abordada por estudos na área [3]. Esta política exige que a quantidade entregue ao cliente complete a capacidade disponível de estoque, reduzindo o número de atendimentos ao longo do horizonte de planejamento. Por fim, propomos um novo algoritmo *branch-and-cut* para resolver o *MDIRPSD* de forma exata, considerando as políticas de estoque ML e OU. Para a versão simplificada do *MDIRPSD* (política ML e sem custo de estoque nos clientes), reportamos uma análise comparativa entre o algoritmo que propomos neste trabalho com método apresentado por [5]. Já para a extensão do *MDIRPSD* com custos de estoque, comparamos também as políticas ML e OU no âmbito do custo logístico global.

O restante do artigo está estruturado como segue. Na seção 2 definimos formalmente o *MDIRPSD* com custos de estoque nos clientes e apresentamos uma formulação para a política OU com entregas fracionadas. Na seção 3 propomos o algoritmo *branch-and-cut*, enquanto a seção 4 reporta os resultados dos experimentos computacionais realizados. A seção 5 tece as conclusões do trabalho.

2. DEFINIÇÃO DO PROBLEMA E MODELAGEM MATEMÁTICA

O *MDIRPSD* é definido sobre um grafo incompleto e não direcionado $G = (\mathcal{V}, \mathcal{E})$, onde o conjunto de vértices $\mathcal{V}$ representa a união do conjunto de Plantas (ou depósitos) $\mathcal{P}$ e Clientes $\mathcal{C}$, enquanto o conjunto $\mathcal{E}$ são as arestas do grafo. Um custo não-negativo de transporte c_{uv} está associado a cada aresta $(u, v) \in \mathcal{E}$, onde $\mathcal{E} = \{(u, v) : u, v \in \mathcal{V} \wedge u, v \oplus \mathcal{P}, u < v\}$.

O horizonte de planejamento é definido pelo conjunto $\mathcal{T}$ com p períodos. Uma frota homogênea de veículos, definida pelo conjunto $\mathcal{K}$ e com capacidade Q por veículo, está disponível para realizar as entregas aos clientes. Cada veículo $k \in \mathcal{K}$ pode ser designado para uma planta j, $j \in \mathcal{P}$, em cada período t, $t \in \mathcal{T}$. Assume-se, sem perda de generalidade, que a capacidade de produção e estoque em cada planta j é ilimitada.

Cada cliente l, $l \in \mathcal{C}$, tem uma demanda determinística não nula, d_l^t a cada período t, com $d_l^0 = 0$. A capacidade máxima de estoque e dada por U_l, sendo constante ao longo de $\mathcal{T}$, enquanto I_l^0 define o nível inicial de estoque, conhecido *a priori*. Por fim, cada cliente l incorre em um custo de estoque h_l por unidade estocada por período. O nível de estoque de cada cliente l, ao final de cada período t, $t \in \mathcal{T} \cup p + 1$, é dado por I_l^t e este não pode ser negativo.

Dois tipos de entregas fracionadas são permitidas a um cliente l no mesmo período t: quando dois ou mais veículos de uma mesma planta j atendem ao referido cliente, bem como quando este mesmo cliente é servido por dois ou mais veículos de plantas distintas. No entanto, a rota de um veículo k deve terminar na mesma planta j de origem.

O objetivo do *MDIRPSD* é minimizar o custo logístico total, dado pelos componentes de estoque e transporte, determinando, para cada período t:

- quando, quanto e a partir de quais plantas cada cliente deve ser atendido;
- quantos veículos designar a cada planta em cada período;
- como combinar as entregas dos clientes em cada período e em cada veículo;

Com relação às variáveis de decisão, cada planta j deve determinar a quantidade q_{jl}^{kt} entregue ao cliente l, com o veículo k no período t. O conjunto restante das variáveis é definido a seguir:

- $Y_{jl}^{kt} = 1$ se o veículo k designado à planta j visita o vértice l no período t, onde $l \in \mathcal{V}$, 0 caso contrário;
- $y_{uv}^{kjt} = 1$ se o veículo k designado à planta j viaja do vértice u ao vértice v no período t, com $(u, v) \in \mathcal{E}, u < v$ e $u, v \oplus \mathcal{P}$, 0 caso contrário;

Um adaptação da variável $Y_{jj}^{kt} = 1$, anui o controle de saída do veículo k da planta j no período t. Finalmente, o *MDIRPSD* é formulado por (1)−(14).

$$\min \sum_{t \in \mathcal{T} \cup \{p+1\}} \sum_{l \in \mathcal{C}} h_l I_l^t + \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{P}} \sum_{k \in \mathcal{K}} \sum_{(u,v) \in \mathcal{E}} c_{uv} y_{uv}^{kjt} \tag{1}$$

subject to

$$I_l^t = I_l^{t-1} + \sum_{j \in \mathcal{P}} \sum_{k \in \mathcal{K}} q_{jl}^{k,t-1} - d_l^t \qquad l \in \mathcal{C}, t \in \mathcal{T} \cup \{p+1\} \tag{2}$$

$$I_l^t \leq U_l \qquad l \in \mathcal{C}, t \in \mathcal{T} \cup \{p+1\} \tag{3}$$

$$I_l^t + \sum_{j \in \mathcal{P}} \sum_{k \in \mathcal{K}} q_{jl}^{kt} \leq U_l \qquad l \in \mathcal{C}, t \in \mathcal{T} \tag{4}$$

$$q_{jl}^{kt} \leq U_l Y_{jl}^{kt} \qquad l \in \mathcal{C}, j \in \mathcal{P}, k \in \mathcal{K}, t \in \mathcal{T} \tag{5}$$

$$\sum_{l \in \mathcal{C}} q_{jl}^{kt} \leq Q Y_{jj}^{kt} \qquad j \in \mathcal{P}, k \in \mathcal{K}, t \in \mathcal{T} \tag{6}$$

$$\sum_{j \in \mathcal{P}} Y_{jj}^{kt} \leq 1 \qquad k \in \mathcal{K}, t \in \mathcal{T} \tag{7}$$

$$\sum_{\substack{u \in \mathcal{V} \\ l<u}} y_{ul}^{kjt} + \sum_{\substack{u \in \mathcal{V} \\ u<l}} y_{lu}^{kjt} = 2Y_{jl}^{kt} \qquad l \in \mathcal{V}, j \in \mathcal{P}, k \in \mathcal{K}, t \in \mathcal{T} \tag{8}$$

$$\sum_{l \in S} \sum_{\substack{u \in S \\ l<u}} y_{lu}^{kjt} \leq \sum_{l \in S} Y_{jl}^{kt} - Y_{jm}^{kt} \qquad S \subseteq \mathcal{C}, |S| \geq 2, m \in S, j \in \mathcal{P}, k \in \mathcal{K}, t \in \mathcal{T} \tag{9}$$

$$I_l^t \geq 0 \qquad l \in \mathcal{C}, t \in \mathcal{T} \cup \{p+1\} \tag{10}$$

$$q_{jl}^{kt} \geq 0 \qquad j \in \mathcal{P}, l \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T} \tag{11}$$

$$Y_{jl}^{kt} \in \{0, 1\} \qquad j \in \mathcal{P}, l \in \mathcal{V}, k \in \mathcal{K}, t \in \mathcal{T} \tag{12}$$

$$y_{jv}^{kjt} \in \{0, 1, 2\} \qquad v \in \mathcal{C}, j \in \mathcal{P}, k \in \mathcal{K}, t \in \mathcal{T} \tag{13}$$

$$y_{uv}^{kjt} \in \{0, 1\} \qquad u, v \in \mathcal{C}, j \in \mathcal{P}, k \in \mathcal{K}, t \in \mathcal{T} \tag{14}$$

A função objetivo 1 minimiza os custos totais, dado pela soma dos custos de estoque e transporte. As restrições 2 garantem a conservação de fluxo, enquanto 3 limitam os níveis de estoques nos clientes. A política *ML* é formulada pelas restrições 4. Já as restrições 5 conectam a quantidade entregue pela planta j ao cliente l com o veículo k se esse cliente for visitado pela tripla (j, k, t). As restrições 6 impedem que a capacidade de um veículo k seja excedida ao atender um conjunto de clientes, enquanto 7 garantem que cada veículo k seja designado a uma única planta j em cada período t. As restrições 8 e 9 estabelecem a conectividade dos vértices e proíbem a formação de sub rotas, respectivamente. O domínio das variáveis de decisão é dado pelas restrições 10−14.

Destacamos que modelo matemático (1)−(14) é suficientemente flexível para representar o problema proposto por [5]. Para isso, basta definir $h_l = 0$, $l \in \mathcal{C}$. Adicionalmente, albergamos a quebra de simetria nas variáveis binárias que representam as decisões de rota ao modelar $y_{jv}^{kjt} = \{0, 1, 2\}$, reduzindo a um único arco as decisões de entregas diretas.

Em nosso melhor conhecimento, a literatura do *IRP* não reporta nenhuma formulação para a política *OU* quando as entregas aos clientes podem ser fracionadas. De maneira a cobrir essa lacuna, definimos uma variável binária adicional Z_l^t, que assume o valor 1 quando o cliente l é visitado ao menos uma vez no período t, e 0 caso contrário. As restrições 15−17 asseguram a política *OU* para o *MDIRPSD*, onde $M = |\mathcal{P}||\mathcal{K}|$ é um *BIG M*.

$$\sum_{j\in\mathcal{P}}\sum_{k\in\mathcal{K}} Y_{jl}^{kt} \leq M Z_l^t \qquad l \in \mathcal{C}, t \in \mathcal{T} \tag{15}$$

$$I_l^t + \sum_{j\in\mathcal{P}}\sum_{k\in\mathcal{K}} q_{jl}^{kt} \geq U_l Z_l^t \qquad l \in \mathcal{C}, t \in \mathcal{T} \tag{16}$$

$$Z_l^t \in \{0, 1\} \qquad l \in \mathcal{C}, t \in \mathcal{T} \tag{17}$$

De maneira a fortalecer os limitantes duais, nós apresentamos um conjunto de desigualdades válidas, propostas por [10, 11], para a variante clássica do *IRP*.

$$y_{jl}^{kjt} \leq 2Y_{jl}^{kt} \qquad j \in \mathcal{P}, l \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T} \tag{18}$$

$$y_{lu}^{kjt} \leq Y_{jl}^{kt} \qquad j \in \mathcal{P}, l, u \in \mathcal{C}, l < u, k \in \mathcal{K}, t \in \mathcal{T} \tag{19}$$

$$y_{lu}^{kjt} \leq Y_{ju}^{kt} \qquad j \in \mathcal{P}, l, u \in \mathcal{C}, l < u, k \in \mathcal{K}, t \in \mathcal{T} \tag{20}$$

$$Y_{jl}^{kt} \leq Y_{jj}^{kt} \qquad j \in \mathcal{P}, l \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T} \tag{21}$$

As desigualdades 18 elevam os limitantes inferiores do modelo, quando um cliente l é atendido por uma entrega direta pelo veículo k designado à planta j no período t. Analogamente, 19 e 20 representam o caso de múltiplos clientes servidos em uma mesma rota. Por fim, as desigualdades 21 asseguram que o cliente l poderá ser atendido pelo veículo k que parte da planta j no período t, somente se esse veículo for utilizado.

Adaptamos as restrições de quebra de simetria apresentadas por [11] para o *IRP* com múltiplos depósitos em 22, que garantem que um veículo de índice k só será utilizado por uma planta j, quando o veículo de índice $k-1$ já estiver em uso por alguma planta.

$$\sum_{j\in\mathcal{P}} Y_{jj}^{kt} \leq \sum_{j\in\mathcal{P}} Y_{jj}^{k-1,t} \qquad t \in \mathcal{T}, k \in \mathcal{K}, k > 1 \qquad (22)$$

Finalmente em 23, consideramos as desigualdades propostas por [4], que calculam o menor intervalo para o atendimento de um cliente, quando a estrutura logística é formada por múltiplos depósitos.

$$\sum_{j\in\mathcal{P}} \sum_{k\in\mathcal{K}} \sum_{t=t_1}^{t_2} Y_{jl}^{kt} \geq \left\lceil \frac{\sum_{t=t_1}^{t_2} d_l^t - U_l}{min\left\{Q, U_l\right\}} \right\rceil \qquad l \in \mathcal{C}, t_1, t_2 \in \mathcal{T}, t_2 > t_1 \qquad (23)$$

3. ALGORITMO *BRANCH-AND-CUT*

Por conta da sua complexidade combinatorial, o número de subconjuntos necessários para gerar todas as restrições de eliminação de subrotas (RES) em 9 é demasiadamente grande, fazendo com que o processo enumerativo pleno seja impraticável. Para superar essa limitação, essas restrições precisam ser geradas e adicionadas ao longo da otimização do problema. Nós delineamos uma abordagem exata para resolver o modelo apresentado na seção 2, onde as RES são adicionadas à arvore de busca, sempre que sub rotas forem identificadas na solução corrente. Esta técnica é conhecida como *Branch-and-Cut* (*B&C*), e combina a clássica metodologia *Branch-and-Bound* (*B&B*) com algoritmos de planos de corte.

Durante a execução do algoritmo *B&B*, a etapa de *branching* ou ramificação é a mais importante. Sempre que existirem variáveis de domínio inteiro com valores fracionários na solução corrente, uma ramificação deve ser imposta. Após uma criteriosa seleção, a variável fracionária escolhida $x = a$, limitar-se-à $x \geq \lceil a \rceil$ e $x \leq \lfloor a \rfloor$. Para o *MDIRPSD*, priorizamos a ramificação das variáveis de saída de um veículo da planta Y_{jj}^{kt}, seguido das variáveis de atendimento de cliente Y_{jl}^{kt} e por último, as variáveis de rotas y_{uv}^{kjt}.

No início do processo de busca, todas as desigualdades válidas são geradas e adicionadas ao nó raiz da árvore do *B&B*. A cada nó resolvido pelo método *B&B*, um algoritmo de busca por RES violadas é então aplicado. Para essa finalidade nós utilizamos o pacote *CVRPSEP*, desenvolvido e disponibilizado por [12], que, ao identificar um RES violada, gera um conjunto de cortes e desigualdades válidas, as quais adaptamos para o *MDIRPSD*. Neste ponto, um novo subproblema é gerado pela ramificação de uma nova variável fracionária selecionada, e o problema original é então reotimizado a partir de um novo nó da árvore de busca. O processo é finalizado após um limite de tempo estabelecido ou ao término da exploração dos nós gerados pela árvore *B&B*.

Devida à complexidade combinatorial das RES o algoritmo *B&C* muitas vezes não é capaz de encontrar soluções incumbentes, especialmente para problemas de médio e grande porte. Para contornar essa limitação, propomos uma estratégia para a obtenção de uma solução factível inicial, partindo de uma reformulação do modelo da seção 2 que desconsidera as variáveis de rota y_{uv}^{kjt}. A nova função objetivo (FO'), dada por 24, considera o custo de entrega direta, caso o cliente seja servido pelo veículo k designado à planta j no período t.

$$\min \sum_{t\in\mathcal{T}\cup\{p+1\}} \sum_{l\in\mathcal{C}} h_l I_l^t + \sum_{j\in\mathcal{P}} \sum_{l\in\mathcal{C}} \sum_{k\in\mathcal{K}} \sum_{t\in\mathcal{T}} c_{jl} Y_{jl}^{kt} \tag{24}$$

A (FO') está sujeito às restrições $(2)-(7) \cup (10)-(12)$, resultando em um modelo de fluxo com entregas diretas. A política OU é igualmente contemplada adicionando as restrições $(15)-(17)$ ao modelo de fluxo.

Para gerar as rotas iniciais e completar a solução incumbente do *MDIRPSD*, resolvemos os *Travel Salesman Problem* (*TSP's*), para as triplas (j, k, t) associadas às variáveis $Y_{jl}^{kt} = 1$, pelo método proposto por [13]. A solução inicial obtida é então inserida no nó raiz do *B&C* proposto. Um esquema para o algoritmo *B&C* é apresentado em 1.

Algorithm 1 Pseudocódigo para o algoritmo *B&C*

```
Obter uma solução inicial e adicionar ao nó raiz da árvore de busca.
No nó raiz da árvore de busca, gerar (2)–(8), (10)–(14) e todas as desigualdades válidas (18)–(23).
Solução do Subproblema: resolver o problema de programação linear (PL) associado ao nó.
Critério de Parada:
if Não existirem mais nós para serem avaliados then
    Pare.
else
    Selecione um novo nó da árvore B&B.
end if
while A solução do PL conter sub rotas do
    Adicione as RES associadas.
    Solução do Subproblema: resolver o PL associado ao nó.
end while
if A solução do PL é inteira then
    Ir para o Critério de Parada.
else
    Ramificação: Escolher e ramificar uma variável fracionária.
    Ir para o Critério de Parada.
end if
```

4. EXPERIMENTOS COMPUTACIONAIS

Os testes computacionais foram realizados em processadores *Core i7*, com 16 GB de memória RAM e sistema operacional Linux 6.6. O algoritmo *B&C*, descrito na seção 3, foi implementado em $C++$ e os modelos de programação linear inteira foram resolvidos com o *solver* Gurobi 8.1.0 com parâmetros *default*. O algoritmo *B&C* foi processado por até 7200 segundos.

O *B&C* foi testado sobre um conjunto de 20 instâncias, adaptadas de [10] para múltiplos depósitos por [5]. Todas as instâncias possuem frota com 3 veículos e o número de plantas depende do número de clientes. Da esquerda para a direita, a Tabela 1 reporta o nome da instância na seguinte estrutura 1n$|\mathcal{C}|$h$|\mathcal{T}|$, onde $|\mathcal{C}|$ é o número de clientes e $|\mathcal{T}|$ o número de períodos. Sequencialmente, são apresentados as soluções (*Upper Bounds - UB*) obtidos pelo *B&C* proposto por [5] com o tempo de processamento na coluna ao lado. As quatro colunas seguintes apresentam os resultados do nosso *B&C*, envolvendo o *UB*, (*Lower Bounds - LB*), o desvio entre eles e o tempo de processamento. A última coluna informa o desvio entre os *UB's*, calculado por $GAP = \frac{UB_{nosso} - UB_{bertazzi}}{UB_{bertazzi}}$, quantificando a comparação entre os métodos. Pela dificuldade inerente ao problema, foi possível encontrar soluções ótimas em apenas 3 instâncias, limitadas à 10 clientes ($1n5h3$, $1n10h3$ e $1n5h6$), enquanto instâncias com 50 clientes apresentam gaps superiores à 40%. Contudo, esse padrão de desempenho é esperado para métodos exatos em problemas dessa natureza [3].

Instancia	Bertazzi el al. (2019)		B&C				GAP
	UB	T(s)	UB	LB	$\frac{UB-LB}{UB}$	T(s)	
1n5h3	1148,80	89,71	1148,80	1148,80	0,00%	1,11	0,00%
1n10h3	2177,99	21700	2110,01	2109,98	0,00%	664,70	-3,12%
1n15h3	4588,88	21700	4013,73	3768,66	6,11%	7200	-12,53%
1n20h3	3263,25	21700	3110,41	2862,78	7,96%	7200	-4,68%
1n25h3	4203,45	21700	3548,92	3183,22	10,30%	7200	-15,57%
1n30h3	5206,49	21700	4342,43	3706,57	14,64%	7200	-16,60%
1n35h3	7013,75	21700	5699,71	4682,80	17,84%	7200	-18,74%
1n40h3	7080,27	21700	5908,26	4702,24	20,41%	7200	-16,55%
1n45h3	7272,02	21700	7088,22	4591,92	35,22%	7200	-2,53%
1n50h3	7881,37	21700	7874,59	5077,56	35,52%	7200	-0,09%
1n5h6	2595,14	8685,15	2595,14	2595,14	0,00%	3,37	0,00%
1n10h6	5102,45	21700	4214,91	3644,21	13,54%	7200	-17,39%
1n15h6	9904,03	21700	8056,82	7376,30	8,45%	7200	-18,65%
1n20h6	11433,40	21700	10519,36	8501,48	19,18%	7200	-7,99%
1n25h6	11949,99	21700	11484,49	8732,21	23,97%	7200	-3,90%
1n30h6	12750,90	21700	12511,56	8670,60	30,70%	7200	-1,88%
1n35h6	12924,00	21700	11347,92	8897,96	21,59%	7200	-12,19%
1n40h6	12208,50	21700	13010,85	9992,68	23,20%	7200	6,57%
1n45h6	10884,90	21700	11019,43	7490,82	32,02%	7200	1,24%
1n50h6	14031,90	21700	14081,32	8260,21	41,34%	7200	0,35%
Média	**7681,07**	**19968,74**	**7184,34**	**5499,81**	**18,10%**	**6155,05**	**-7,21%**

Tabela 1: Resultados comparativos com o *B&C* proposto por [5]

A análise comparativa entre algoritmos apresenta dados interessantes. Salientamos que os resultados de [5] não reportam os *Lower Bounds* (LB) e o que o tempo de processamento foi de 21000 segundos, muito superior aos 7200 que adotamos. Embora a diferença de implementação, *solver* e hardware não permitam comparações justas, podemos observar em 17 das 20 instâncias o UB obtido pelo nosso algoritmo foi melhor que o **UB** dos autores comparados. Além disso, conseguimos provar a otimalidade para a instância $1n10h3$ em 664,7 segundos, ao passo que os [5] não o fez em 21000 segundos. A diferença no tempo computacional pode ser explicada pela solução inicial fornecida, o que pode direcionar o esforço do método para provar a otimalidade, em detrimento ao processo de obtenção de uma solução factível. De forma geral, conseguimos reduzir os custos em $7,21\%$ em média.

O segundo grupo de análise compara as políticas de estoque ML e OU sem considerar os custos de manutenção, ou seja, $h_l = 0$, $l \in \mathcal{C}$. Da esquerda para a direita a Tabela 2 apresenta o nome das instâncias, os resultados da política ML para UB, LB, desvio e tempo de processamento em segundos, seguido das mesmas informações para a política OU. A última coluna compara os UB's entre as políticas, calculada por $GAP = \frac{UB_{OU}-UB_{ML}}{UB_{ML}}$. A dificuldade na resolução das políticas foi semelhante no âmbito do tempo de processamento. Em termos de solução, o *B&C* para a política OU encontrou uma solução ótima a mais que a política ML (instância $1n15h3$), devido à redução do número de visitas. O desempenho geral do algoritmo *B&C* foi semelhante entre as políticas, com tempo médio global e desvio médio entre UB e LB muito próximos. Já pelo aspecto do custo total, a política OU é em média $6,47\%$ mais custosa, mesmo sem considerar o custo de estocagem nos clientes. Esse padrão de resposta é condizente com a formulação do problema, pois, em último caso, a política ML recai na política OU.

Por fim, comparamos as duas políticas para a formulação apresentada na seção 2,

Instância	Política ML				Política OU				GAP
	UB	LB	$\frac{UB-LB}{UB}$	T(s)	UB	LB	$\frac{UB-LB}{UB}$	T(s)	
1n5h3	1148,80	1148,80	0,00%	1,11	1177,69	1177,69	0,00%	0,51	2,51%
1n10h3	2110,01	2109,98	0,00%	664,70	2298,24	2298,24	0,00%	646,03	8,92%
1n15h3	4013,73	3768,66	6,11%	7200	4173,13	4173,44	0,01%	7112,54	3,98%
1n20h3	3110,41	2862,78	7,96%	7200	3313,31	3121,11	5,80%	7200	6,52%
1n25h3	3548,92	3183,22	10,30%	7200	3677,37	3595,77	2,22%	7200	3,62%
1n30h3	4342,43	3706,57	14,64%	7200	4397,71	4082,52	7,17%	7200	1,27%
1n35h3	5699,71	4682,80	17,84%	7200	7340,37	5155,16	29,77%	7200	28,79%
1n40h3	5908,26	4702,24	20,41%	7200	7429,49	5126,49	31,00%	7200	25,75%
1n45h3	7088,22	4591,92	35,22%	7200	7351,31	4709,15	35,94%	7200	3,71%
1n50h3	7874,59	5077,56	35,52%	7200	8622,25	5216,66	39,50%	7200	9,49%
1n5h6	2595,14	2595,14	0,00%	3,37	2607,74	2607,74	0,00%	2,83	0,49%
1n10h6	4214,91	3644,21	13,54%	7200	4497,83	3994,43	11,19%	7200	6,71%
1n15h6	8056,82	7376,30	8,45%	7200	8438,64	7474,66	11,42%	7200	4,74%
1n20h6	10519,36	8501,48	19,18%	7200	12312,43	8940,64	27,39%	7200	17,05%
1n25h6	11484,49	8732,21	23,97%	7200	11099,11	8749,27	21,17%	7200	-3,36%
1n30h6	12511,56	8670,60	30,70%	7200	13153,27	8450,91	35,75%	7200	5,13%
1n35h6	11347,92	8897,96	21,59%	7200	11295,75	8541,92	24,38%	7200	-0,46%
1n40h6	13010,85	9992,68	23,20%	7200	13348,46	10293,58	22,89%	7200	2,59%
1n45h6	11019,43	7490,82	32,02%	7200	11221,44	8489,99	24,34%	7200	1,83%
1n50h6	14081,32	8260,21	41,34%	7200	14086,57	8750,18	37,88%	7200	0,04%
Média	**7184,34**	**5499,81**	**18,10%**	**5747,46**	**7592,12**	**6155,05**	**18,39%**	**6149,86**	**6,47%**

Tabela 2: Resultados comparativos para as políticas *OU* e *ML* sem considerar os custos de estocagem

considerando os custos de estocagem nos clientes. A Tabela 3 reporta a mesma estrutura da tabela anterior. Com relação ao desempenho do algoritmo *B&C*, verificamos que as soluções ótimas encontradas são as mesmas para as duas políticas, com desvios globais médios semelhantes para o tempo de processamento e os resultados dos limitantes *UB* e *LB*. Já a análise comparativa destaca a importância dos custos de estocagem. Como a política *OU* mantém níveis de estoque mais elevados nos clientes, a comparação entre as políticas é mais díspar. Em valores médios para o conjunto das 20 instâncias testadas, a política *OU* é $11,95\%$ mais custosa que a política *ML*. Esse dado é condizente com os resultados reportados por [10, 14, 4], que resolvem diferentes variações do *Inventory Routing Problem* sob a política de estoque *OU*.

Instância	Política ML				Política OU				GAP
	UB	LB	$\frac{UB-LB}{UB}$	T(s)	UB	LB	$\frac{UB-LB}{UB}$	T(s)	
1n5h3	1278,11	1278,11	0,00%	0,74	1378,28	1378,28	0,00%	0,44	7,84%
1n10h3	2671,71	2671,71	0,00%	466,45	3050,06	3050,06	0,00%	198,89	14,16%
1n15h3	4588,73	4501,15	1,91%	7200	5035,57	5035,57	0,00%	6019,75	9,74%
1n20h3	3938,59	3832,00	2,71%	7200	4513,63	4300,28	4,73%	7200	14,60%
1n25h3	4512,84	4200,14	6,93%	7200	5035,51	4882,98	3,03%	7200	11,58%
1n30h3	5756,19	5435,67	5,57%	7200	6726,83	6394,82	4,94%	7200	16,86%
1n35h3	7008,24	6204,91	11,46%	7200	8573,82	7146,71	16,64%	7200	22,34%
1n40h3	7969,30	6748,08	15,32%	7200	9835,01	7838,64	20,30%	7200	23,41%
1n45h3	9083,14	6669,32	26,57%	7200	9942,74	7412,93	25,44%	7200	9,46%
1n50h3	10282,09	7311,93	28,89%	7200	11513,42	7876,56	31,59%	7200	11,98%
1n5h6	3023,43	3023,43	0,00%	3,02	3238,46	3238,46	0,00%	2,17	7,11%
1n10h6	5111,14	4497,20	12,01%	7200	5576,66	5162,65	7,42%	7200	9,11%
1n15h6	9267,71	8472,42	8,58%	7200	9880,86	8995,69	8,96%	7200	6,62%
1n20h6	11860,02	9944,18	16,15%	7200	14953,22	10947,82	26,79%	7200	26,08%
1n25h6	11704,31	10114,42	13,58%	7200	14016,66	10675,82	23,83%	7200	19,76%
1n30h6	15528,29	11331,69	27,03%	7200	16942,74	12055,63	28,84%	7200	9,11%
1n35h6	13799,01	11140,03	19,27%	7200	14285,10	11417,87	20,07%	7200	3,52%
1n40h6	16967,71	13221,60	22,08%	7200	17981,20	14346,50	20,21%	7200	5,97%
1n45h6	14660,56	11175,57	23,77%	7200	15690,00	12538,76	20,08%	7200	7,02%
1n50h6	18676,15	11302,30	39,48%	7200	19167,84	12909,04	32,65%	7200	2,63%
Média	**8884,36**	**7153,79**	**14,07%**	**6143,51**	**9866,88**	**7880,25**	**14,78%**	**6071,06**	**11,95%**

Tabela 3: Resultados comparativos para as políticas *OU* e *ML* considerando custos de estocagem

5. CONCLUSÕES

Neste trabalho abordamos um problema logístico prático, que emerge do paradigma do estoque gerenciado pelo fornecedor um em logística urbana, denominado *Multi Depot Inventory Routing Problem with Split Deliveries* (*MDIRPSD*) ou Problema de Roteirização e Estoques com Múltiplos Depósitos e Entregas Fracionadas. A partir do trabalho de [5] que introduziu o problema na literatura, nós expandimos a formulação e incorporamos os custos de estocagem no cliente. Além disso, adaptamos a política de estoque *OU* para o *MDIRPSD* e desenvolvemos um novo algoritmo *B&C* com um mecanismo de obtenção de uma solução inicial, além de algumas alterações estruturais. Os resultados computacionais apontam que nosso método é superior à abordagem proposta por [5], sendo capaz de reduzir os *UB's* conhecidos, além de provar a otimalidade para uma instância em aberta. A análise comparativa entre as políticas de estoque *ML* e *OU* demonstram que níveis mais elevados de estoque geram soluções mais custosas, e destacam a importância de se considerar os custos de estoque no processo decisório.

6. AGRADECIMENTOS

Agradecemos ao Centro de Computação Científica e Software Livre (C3SL) da Universidade Federal do Paraná pela assistência e recursos computacionais oferecidos. Agradecemos à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), pelo apoio financeiro N. 1554767.

7. REFERÊNCIAS BIBLIOGRÁFICAS

[1] ANDERSSON, H. et al. Industrial aspects and literature survey: Combined inventory management and routing. v. 37, n. 9, p. 1515–1536, 2010. 2

[2] GOVINDAN, K. Vendor-managed inventory: A review based on dimensions. *International Journal of Production Research*, v. 51, n. 13, p. 3808–3835, 2013. 2

[3] COELHO, L. C.; CORDEAU, J.-F.; LAPORTE, G. Thirty Years of Inventory Routing. *Transportation Science*, v. 48, n. 1, p. 1–19, 2013. 2, 3, 7

[4] GUIMARÃES, T. A. et al. The two-echelon multi-depot inventory-routing problem. *Computers and Operations Research*, v. 101, p. 220–233, 2019. 2, 6, 9

[5] BERTAZZI, L. et al. A matheuristic algorithm for the multi-depot inventory routing problem. *Transportation Research Part E: Logistics and Transportation Review*, v. 122, p. 524–544, 2019. 2, 3, 5, 7, 8, 10

[6] COELHO, L. C.; LAPORTE, G. A branch-and-cut algorithm for the multi-product multi-vehicle inventory-routing problem. *International Journal of Production Research*, v. 51, n. 23-24, p. 7156–7169, 2013. 2

[7] COELHO, L. C.; LAPORTE, G. The exact solution of several classes of inventory-routing problems. *Computers and Operations Research*, v. 40, n. 2, p. 558–565, 2013. 2

[8] SERNA, M. D. A.; CORTES, J. A. Z.; SEPULVEDA, D. G. Modeling the Inventory Routing Problem (IRP) with multiple depots with genetic algorithms. *IEEE Latin America Transactions*, v. 13, n. 12, p. 3959–3965, 2015. 2

[9] BERTAZZI, L.; PALETTA, G.; SPERANZA, M. G. Deterministic Order-Up-To Level Policies in an Inventory Routing Problem. *Transportation Science*, v. 36, n. 1, p. 119–132, 2003. 3

[10] ARCHETTI, C. et al. A branch-and-cut algorithm for a vendor-managed inventory-routing problem. *Transportation Science*, v. 41, n. 3, p. 382–391, 2007. 5, 7, 9

[11] COELHO, L. C.; LAPORTE, G. Improved solutions for inventory-routing problems through valid inequalities and input ordering. *International Journal of Production Economics*, Elsevier, v. 155, p. 391–397, 2014. 5

[12] LYSGAARD, J.; LETCHFORD, A. N.; EGLESE, R. W. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, v. 100, n. 2, p. 423–445, 2004. 6

[13] PADBERG, M.; RINALDI, G. A Branch-and-Cut Algorithm for the Resolution of Large-Scale Symmetric Traveling Salesman Problems. *SIAM Review*, v. 33, n. 1, p. 60–100, 1991. 7

[14] COELHO, L. C.; CORDEAU, J. F.; LAPORTE, G. Consistency in multi-vehicle inventory-routing. *Transportation Research Part C: Emerging Technologies*, v. 24, p. 270–287, 2012. 9