HEURÍSTICA HÍBRIDA APLICADA AO PROBLEMA DE ESTOQUE E ROTEAMENTO DE MÚLTIPLOS VEÍCULOS COM UM DEPÓSITO E UM PRODUTO

Pierre Novis Mendonça (pierremendonca@petrobras.com.br)
Mayron Rodrigues de Almeida (mra@petrobras.com.br)
Petrobras - Gerência de Soluções de Pesquisa Operacional
Av. Nilo Peçanha, 151 - Centro - Rio de Janeiro - RJ

RESUMO. Este artigo trata do Problema de Estoque e Roteamento de Múltiplos Veículos (PERMV). Este problema engloba o gerenciamento do estoque do cliente, o estabelecimento da frequência e quantidade de produto entregue, além do roteiro percorrido pelo veículo ao longo de um horizonte de planejamento, possibilitando uma redução simultânea dos custos de estoque e transporte. O algoritmo foi avaliado a partir de experimentos computacionais em um conjunto de instâncias da literatura e os resultados obtidos validam sua eficiência.

PALAVRAS-CHAVE: ILS, MIRP, Algoritmo Híbrido

Área principal. L&T - Logística e Transportes

ABSTRACT. This paper deal with the Multi-vehicle Inventory Routing Problem (MIRP). This problem includes the management of customer inventory, determine the frequency and quantity delivered to the customer in addition the vehicle routing over a planning horizon, while reducing the costs of storage and distribution. The main objective of the work is propose a general procedure using hibrid Metaheuristic in order to solve real problems.The proposed algorithm was evaluated from computational experiments with a set of instances from the literature and the results have showed its effectiveness.

KEYWORDS. MIRP, ILS, Hibrid Algorithm

Main area. L&T - Logistics and Transport

1 Introdução

Atualmente, a cadeia de suprimentos vem cumprindo um papel importante no desempenho das empresas que estão percebendo a importância da melhoria do nível de serviço de atendimento aos clientes através da logística. Neste contexto, destaca-se o *Vendor Managed Inventory* (VMI) ou Estoque Gerenciado pelo Fornecedor (EGF). Segundo [Campbell et al.(1998)] o EGF cria valor tanto para o fornecedor quanto para o cliente, em uma situação de ganha-ganha, representado pela redução de custos de distribuição e melhor gerenciamento das entregas de um lado, e pela eliminação da necessidade de controle de estoques do outro. A implantação de um sistema tipo EGF requer a resolução de um complexo problema de otimização denominado Inventory Routing Problem (IRP) ou Problema de Roteirização e

Estoques (PRE), que combina o gerenciamento de estoque com programação de entregas e roteirização de veículos.

Desde que este problema foi proposto por [Bell et al.(1983)] para um caso de distribuição de gás industrial, diversos trabalhos sobre diversas variantes do problema e várias abordagens tem sido propostas. Geralmente o problema é classificado seguindo os critérios que envolvem horizonte de tempo, política de reposição, composição e tamanho da frota, disponibilidade de informação sobre a demanda, roteiros, entre outros. Com destaque, a política de reposição define as regras principais sobre a quantidade entregue aos clientes, impactando diretamente no custo total. Mais comumente, reporta-se na literatura a política *Maximum Level* (ML), que flexibiliza a quantidade entregue ao cliente, estando limitada à sua disponibilidade de armazenamento no período e O*rder-up-to* (OU), que determina ao fornecedor entregar a máxima quantidade possível, sempre que um cliente for visitado. Para uma referência abrangente sobre o assunto ver [Coelho et al.(2013)].

Para o caso particular de apenas um veículo, diversas heurísticas ([Archetti et al.(2012)], [Bertazzi et al.(2002)] e [Coelho(2012)]) e um algoritmo exato ([Archetti et al.(2007)]) foram propostas. Para o caso de múltiplos veículos, [Coelho et al.(2012), Coelho(2012)] propõe um algoritmo exato *branch-and-cut* e uma metaheurística ALNS (*Adaptative Large Neighborhood Search*) considerando frota homogênea e heterogênea.

Este trabalho busca definir um procedimento geral para o tratamento de problemas reais através da implementação de metaheurísticas. As dificuldades comuns são: o tratamento de muitas restrições e a manutenção do código. A escolha do IRP deu-se por ser este um problema que apresenta uma caracterísitca interessante: dificuldade de se encontrar uma solução viável através de heurísticas.

2 Descrição do problema

O problema tratado neste trabalho considera a necessidade de se entregar um tipo de produto a partir de um fornecedor a diversos clientes ao longo do tempo. Tanto os estoques dos clientes quanto o do fornecedor tem custos e devem permanecer entre limites mínimos e máximos. Todos os estoques iniciais são conhecidos. A produção do fornecedor e o consumo de cada cliente são constantes em cada período. Entregam-se os produtos por meio de uma frota limitada e homogênea. As distâncias entre clientes e entre clientes e fornecedor são conhecidas e o custo do transporte é proporcional à distância percorrida pelos veículos.

Uma solução desse problema deve indicar para cada período: a quantidade de produto a ser entregue para cada cliente e por quais veículos, a rota de cada veículo e o custo total. A descrição formal do problema é a mesma feita em [Coelho(2012)] com as diferenças de que os veículos tem a mesma capacidade e não há impedimento para que um cliente seja visitado por mais de um veículo no mesmo período. Nas tabelas 1, 2 e 3 encontram-se as descrições dos índices, parâmetros e variáveis usados na modelagem. O problema de programação matemática é apresentado no modelo 1. A função objetivo tem duas parcelas principais, a soma dos custos de estoques e a soma dos custos de transporte. As restrições 2 e 3 calculam o nível de estoque no forcecedor, respectivamente, no final do período 0 e no final dos períodos subsequentes. As restrições 4 e 5 são como as anteriores mas referem-se aos clientes. As restrições 6 e 7 determinam os limites mínimo e máximo dos estoques. A carga levada por um veículo em um período é igual à soma das cargas entregues aos clientes neste período, garantido pela restrição 8. A restrição 9 limita a capacidade de carga de cada veículo. A restrição 10 indica que se uma entrega é feita, esta deve ser, no máximo, igual à capacidade do veículo ou zero em caso contrário. As restrições 11 a 13 determinam que sempre que um cliente é visitado devem haver exatamente um

nó antecessor e um nó sucessor. As retrições 14 a 17 garantem que a solução não contenha múltiplos ciclos para cada veículo. As restrições 18 e 19 são restrições de integralidade e, finalmente, a restrição 20 garante que as quantidades entregues sejam não-negativas.

Tabela 1 – Índices

Símbolo	Descrição
d	período
i, j	local
v	veículo

Tabela 2 – Parâmetros Dados

Símbolo	Unidade	Descrição
Ce_i	$\$/u$	custo unitário do estoque em i
Cr_{ij}	\$	custo unitário de transporte entre i e j
$\overline{E}_i$	u	estoque máximo em i
$\underline{E}_i$	u	estoque mínimo em i
Ei_i	u	estoque em i no início da programação
$\overline{L}$	u	capacidade máxima do veículo
Nd	1	número de períodos
NS	1	número de soluções na memória
Qc_i	u	consumo periódico em i
Qf	u	produção periódica

Tabela 3 – Variáveis

Símbolo	Unidade	Descrição
b_{dijv}	1	indica que o veículo v percorre o trecho ij no período d
e_{di}	u	estoque em i no final do período d
l_{dv}	u	carga levada em v no período d
q_{div}	u	entrega em i no período d por v
w_{div}	u	soma das entregas até i no período d feitas por v
y_{div}	1	indica entrega em i no período d por v

3 Algoritmo proposto

O algoritmo implementado é uma heurística ILS (Iterated Local Search) com pequenas alterações em relação ao descrito por Lourenço et al. [Gendreau and Potvin(2010)]. O arcabouço geral é mostrado no algoritmo 1. A principal diferença é o uso de uma operação de combinação de duas soluções como uma forma de perturbação.

Embora seja possível trabalhar tanto com soluções viáveis quanto inviáveis, optou-se por trabalhar apenas com as soluções viáveis porque testes preliminares indicaram que bons resultados são encontrados em muito menos tempo e a implementação da função objetivo é mais simples.

Modelo 1 – Programação matemática

$$\min \quad \sum_{di} Ce_i \cdot e_{di} + \sum_{dijv} Cr_{ij} \cdot b_{dijv} \tag{1}$$

$$s.a \quad e_{di} = Ei_i + Qf - \sum_{jv|j\neq 0} q_{djv} \qquad \{(d,i)|(d=0)\wedge(i=0)\} \tag{2}$$

$$e_{di} = e_{d-1,i} + Qf - \sum_{jv|j\neq 0} q_{djv} \qquad \{(d,i)|(d>0)\wedge(i=0)\} \tag{3}$$

$$e_{di} = Ei_i - Qc_i + \sum_{v} q_{div} \qquad \{(d,i)|(d=0)\wedge(i>0)\} \tag{4}$$

$$e_{di} = e_{d-1,i} - Qc_i + \sum_{v} q_{div} \qquad \{(d,i)|(d>0)\wedge(i>0)\} \tag{5}$$

$$\underline{E}_i \leq e_{di} \qquad d \in D, i \in I \tag{6}$$

$$e_{di} \leq \overline{E}_i - Qc_i \qquad d \in D, i \in I \tag{7}$$

$$l_{dv} = \sum_{i} q_{div} \qquad d \in D, v \in V \tag{8}$$

$$l_{dv} \leq \overline{L} \qquad d \in D, v \in V \tag{9}$$

$$q_{div} \leq y_{div} \cdot \overline{L} \qquad d \in D, i \in I\backslash\{0\}, v \in V \tag{10}$$

$$\sum_{j} b_{dijv} = y_{div} \qquad d \in D, i \in I\backslash\{0\}, v \in V \tag{11}$$

$$\sum_{j} b_{djiv} = y_{div} \qquad d \in D, i \in I\backslash\{0\}, v \in V \tag{12}$$

$$\sum_{j|j\neq 0} b_{dijv} \leq 1 \qquad d \in D, i = 0, v \in V \tag{13}$$

$$w_{div} - w_{djv} + \overline{L} \cdot b_{dijv} \leq \overline{L} - q_{djv} \qquad d \in D, i, j \in I\backslash\{0\}, v \in V \tag{14}$$

$$q_{div} \leq w_{div} \qquad d \in D, i \in I\backslash\{0\}, v \in V \tag{15}$$

$$0 \leq w_{div} \qquad d \in D, i \in I\backslash\{0\}, v \in V \tag{16}$$

$$w_{div} \leq \overline{L} \qquad d \in D, i \in I\backslash\{0\}, v \in V \tag{17}$$

$$b_{dijv} \in \{0,1\} \qquad d \in D, i, j \in I, v \in V \tag{18}$$

$$y_{div} \in \{0,1\} \qquad d \in D, i \in I, v \in V \tag{19}$$

$$0 \leq q_{div} \qquad d \in D, i \in I\backslash\{0\}, v \in V \tag{20}$$

```
1  t = 0
2  memoria = iniciacao(NS)
3  enquanto t < T faça
4  |  s = memoria(rand(1,NS))
5  |  se rand() < 0,4 então
6  |  |  s' = combina(s, memoria(rand(1,NS)))
7  |  senão
8  |  |  s' = perturba(s)
9  |  fim
10 |  s'' = buscaLocal(s')
11 |  se s'' < memoria(NS) então
12 |  |  insere(s'', memoria)
13 |  |  ordena(memoria)
14 |  fim
15 fim
```

Algoritmo 1 – Arcabouço geral

Constata-se que uma solução do problema pode ser construída a partir de apenas duas informações básicas: a variável q_{div} e por listas que representem a rota de cada veículo. Neste trabalho a variável r_{dv} refere-se à rota do veículo v no período d e é implementada como uma lista encadeada cujos primeiro e último nós tem valor 0, indicando o inicio e o fim do percurso no local do fornecedor. Entre o segundo e o penúltimo nós estão os nós dos clientes visitados.

3.1 Iniciação

A iniciação aleatória é uma opção interessante a fim de se obter diversidade entre as soluções. Sendo conhecidos o período, o cliente e o veículo de cada entrega, uma rota pode ser determinada para cada veículo. A forma de se implementar essa iniciação é sortear quantidades entre 0 e a capacidade do veículo:

$$Sq_{div} = \text{rand}(0, \overline{L}),\ d \in D, i \in I \backslash \{0\}, v \in V$$

Determinadas todas as quantidades Sq_{div}, uma solução viável é encontrada em duas etapas: a primeira pela solução de um problema de programação quadrática e depois pela determinação de cada uma das rotas. O modelo quadrático é obtido modificando-se o modelo 1 da seguinte maneira: retiram-se as restrições 10 a 19 e substitui-se a função objetivo pela função 21. Este modelo relaxa as retrições de roteamento e garante que os estoques do depósito, os estoques dos clientes e os limites de carga dos veículos sejam respeitados.

$$\min \sum_{div} (Sq_{div} - q_{div})^2 \tag{21}$$

Em seguida, uma rota é definida para cada veículo em cada período incluindo-se, em ordem aleatória, o índice de cada cliente na lista r_{dv}:

$$r_{dv} = \{0\} \bigcup \{i | q_{div} > 0\}$$

Por fim, uma operação de busca local é executada e essa solução é incluída na memória. A operação de iniciação é repetida NS vezes.

3.2 Perturbação

À medida em que as soluções vão sendo melhoradas, observa-se que é preciso mudar a forma de perturbá-las a fim de se evitar que retornem, depois da busca local, aos seus pontos originais. No entanto, perturbações muito fortes frequentemente geram soluções que não se tornam melhores que aquelas anteriores à perturbação. Uma forma de se obter perturbações bem ajustadas é implementar algum tipo de algoritmo adapatativo ou implementar variadas perturbações. Neste trabalho, há três formas diferentes de se perturbar uma solução: aleatória, junção de duas rotas e união de entregas de mesmo cliente.

Todos os tipos de perturbação consistem em se modificar as variáveis q_{div} e, em seguida, tornar a solução viável. O algoritmo 2 descreve a operação de perturbação.

1 $o = \text{rand}(0,2)$
2 **se** $o = 0$ **então** perturbacaoAleatória()
3 **se** $o = 1$ **então** perturbacaoUneRotas()
4 **se** $o = 2$ **então** perturbacaoUneEntregas()
5 $Y_{div} = 1, \{(d,i,v) | q_{div} > 0\}$
6 $SY = \sum_{div} Y_{div}; CE = \sum_{di} Ce_i \cdot e_{di}$
7 $K1 = \text{rand}(0.0,1.0); K2 = \text{rand}(0.0,1.0); K3 = \text{rand}(0.0,1.0)$
8 $K1 = \frac{K1}{K1+K2+K3}$
9 $K2 = \frac{K2}{K1+K2+K3}$
10 $K3 = \frac{K3}{K1+K2+K3}$
11 resolveMIP
12 atualizaRotas()
13 avalia()

Algoritmo 2 – Perturbação

Os diferentes tipos de pertubações são:

Aleatória todas as entregas são alteradas aleatoriamente. Cada variável q_{div} tem uma probabilidade de 0.1 de ser eliminada, de 0.1 de ser aumentada até a capacidade do veículo e de 0.8 de ser modificada em $\pm 4\%$.

União de rotas dois veículos em um mesmo período são escolhidos aleatoriamente. As entregas dos clientes atendidos apenas pelo primeiro veículo são inseridos no fim da rota do segundo veículo. Os clientes atendidos pelos dois veículos tem as entregas somadas no segundo veículo.

União de entregas dois periodos com um intervalo de, no máximo, dois dias são escolhidos aleatoriamente. Todos os clientes que tem entregas em ambos os períodos passam a ter a soma das entregas somente em um dos períodos. Esta operação pode ser realizada mantendo-se as entregas do primeiro ou do segundo período e eliminado-se o outro.

O retorno à viabilidade é feito por meio da solução de um problema de programação inteira mista tendo como base o modelo 1. O problema deve ser modificado substituindo-se as restrições 11 a 18 pelas restrições 23, 24 e 25 e substituindo-se a função objetivo pela função 22.

$$K1\frac{sdy}{SY} + K2\frac{ce}{CE} + K3\frac{sy}{SY} \tag{22}$$

$$ce = \sum_{di} Ce_i \cdot e_{di} \tag{23}$$

$$sy = \sum_{div} y_{div} \tag{24}$$

$$sdy = \sum_{div|i>0} Y_{div} + (1 - 2 \cdot Y_{div}) y_{div} \tag{25}$$

Uma vez que os novos valores das variáveis q_{div} estão definidos, as listas de rotas r_{dv} são atualizadas. A atualização ocorre quando há mudança no atendimento de cada cliente. São duas as alternativas: na solução original o cliente i era atendido e passou a não ser atendido depois da perturbação, então o nó i é removido da rota. Contrariamente, o nó i é inserido por meio de um algoritmo de melhor inserção.

Finalmente, a função objetivo 1 é avaliada.

3.3 Combinação

A combinação de duas soluções diferentes é uma forma de se fazer uma perturbação misturando-se as características das soluções originais. Neste trabalho, a combinação é feita definindo-se o valor de cada entrega q_{div} como igual ao da primeira solução ou igual ao da segunda solução. Essa escolha é feita com uma probabilidade, que varia entre 0.01 e 0.2, definida no início da execução dessa operação. Determinados os novos valores de q_{div} realizam-se todos os passos a partir da linha 5 do algoritmo 2.

3.4 Busca local

A operação de busca local é realizada por meio da aplicação da técnica *Randomized Variable Neighborhood Descent* (RVND) [Penna(2013), Penna et al.(2013)a, Penna et al.(2013)b, Subramanian(2012), Subramanian et al.(2012), Subramanian et al.(2013)], na qual diversas estruturas de vizinhança são exploradas exaustivamente. As vizinhanças empregadas são as seguintes: **2-opt** (dois arcos presentes em uma rota são substituídos por dois arcos ausentes que restabeleçam sua viabilidade); **Exchange** (dois clientes de uma rota tem suas posições trocadas); **Reinsertion** (a posição de um cliente na rota é alterada); **Cross** (duas rotas são divididas em duas partes cada uma. O ínicio da primeira rota é ligada ao término da segunda e o ínicio da segunda, ao término da primeira. Esta operação é testada para todos os pares de rotas de um mesmo período, para todos os períodos); **Shift(1, 0)** (uma entrega deixa de ser feita e outra entrega pode ser modificada. Ou seja, uma variável q_{div} é zerada e uma variável q_{hju} pode ter seu valor alterado); **Swap(1, 1)** (duas entregas deixam de ser feitas e duas outras podem ter seus valores modificados); **Swap(2, 2)** (dois clientes de uma mesma rota são transferidos para uma segunda rota do mesmo período e dois outros clientes desta rota são transferidos para a primeira) e **MelhorQ** (a quantidade de cada entrega, q_{div}, é maximizada ou minimizada de acordo com sua influência na função objetivo, mantendo-se a viabilidade da solução).

As vizinhanças *2-opt*, *Exchange* e *Reinsertion* são operações intra-rotas e apresentam a interessante característica de não exigir verificação de viabilidade da solução, sendo bastante verificar a variação da função objetivo. Cada uma dessas vizinhanças são experimentadas para todos os veículos em todos os períodos. Neste trabalho, é relevante destacar que as demais vizinhanças foram implementadas de uma forma não usual no que diz respeito ao tratamento das restrições. A garantia de que uma solução se mantém viável é feita resolvendo-se as equações e inequações relacionadas à operação em questão.

1 $feas = 1$
2 **para cada** d **faça**
3 **para cada** v **faça**
4 **para cada** $u \neq v$ **faça**
5 **para cada** $i > 0$ **faça**
6 **para cada** $j | (j \neq i) \wedge (j > 0)$ **faça**
7 $\overline{q} = \infty$
8 $\underline{q} = -\infty$
9 $\overline{q} = \min(\overline{q}, -l_{du} + \overline{L})$
10 $\underline{q} = \max(\underline{q}, -q_{dju})$
11 **para** $y \in \{d, ..., Nd-1\}$ **faça**
12 $\overline{q} = \min(\overline{q}, \overline{E}_j - e_{yj} - Qc_{yj})$
13 $\underline{q} = \max(\underline{q}, \underline{E}_j - e_{yj})$
14 $\overline{q} = \min(\overline{q}, e_{y0} + q_{div} - \underline{E}_0)$
15 $\underline{q} = \max(\underline{q}, e_{y0} + q_{div} - \overline{E}_0)$
16 **se** $\underline{E}_i > e_{yi} - q_{div}$ **então** $feas = 0$
17 **fim**
18 **se** $Ce_j > Ce_0$ **então** $q = \underline{q}$
19 **senão** $q = \overline{q}$
20 $\Delta ce = (Nd - d) \cdot [(Ce_j - Ce_0) \cdot q + (Ce_0 - Ce_i) \cdot q_{div}]$
21 $\Delta crv = \text{calculaRemocao}(i, r_{dv})$
22 $\Delta cru = \text{calculaMelhorInserção}(q_{dju} + q, j, r_{du})$
23 **se** $(\Delta ce + \Delta crv + \Delta cru < 0) \wedge (feas = 1)$ **então**
24 $l_{dv} = l_{dv} - q_{div}$
25 $l_{du} = l_{du} + q$
26 **para** $y \in \{d, ..., Nd-1\}$ **faça**
27 $e_{yi} = e_{yi} - q_{div}$
28 $e_{yj} = e_{yj} + q$
29 $e_{y0} = e_{y0} + q_{div} - q$
30 **fim**
31 $q_{div} = 0$
32 executaRemoção(i, r_{dv})
33 $q_{dju} = q_{dju} + q$
34 executaMelhorInserção(q_{dju}, j, r_{du})
35 **fim**
36 **fim**
37 **fim**
38 **fim**
39 **fim**
40 **fim**

Algoritmo 3 – Shift(1,0)

Considere-se, como exemplo, a operação *Shift(1, 0)* que pode transferir uma entrega no período d a um cliente i pelo veículo v a um cliente j pelo veículo u no mesmo período. Portanto, no caso da função objetivo diminuir, a operação é aceita e tem-se $q_{div} = 0$ e $q_{dju} \geq 0$. Sejam as variáveis $\overline{q}$ e $\underline{q}$, os acréscimos máximo e mínimo à entrega q_{dju}, o algoritmo 3 descreve a busca local na vizinhança *Shift(1,0)* para a situação de transferência de entrega em mesmo período, clientes diferentes e veículos diferentes.

Entre as linhas 7 e 17, as alterações viáveis da entrega q_{dju} são calculados. O aumento máximo de

carga do veículo u é calculado na linha 9 e corresponde à restrição 9 do modelo 1. A linha 10 calcula a restrição 20. As restrições de estoque 6 e 7 só precisam ser calculadas entre os dias d e $Nd-1$. As linhas 12 e 13 garantem que o estoque em j não ultrapassará seus limites máximo e mínimo respectivamnte. As linhas 14 e 15 garantem que o estoque no fornecedor também não ultrapassará seus limites. Finalmente, a linha 16 identifica que a entrega ao cliente i pode ser eliminada sem violar a restrição de estoque mínimo. A variação do custo do estoque é caculado na linha 20. Quando o custo do estoque em j é maior que o custo do estoque no fornecedor é interessante se minimizar a entrega a j e vice-versa, então a decisão de se aumentar ou diminuir a entrega em j é determinada na linha 18. O custo de se remover i da rota r_{dv}, Δcrv, é calculado na linha 21. A função calculaMelhorInserção() avalia a diferença na função objetivo em quatro situações:

- $q_{djv}+q=0$ e j não faz parte de r_{du}, então uma entrega a j não é feita e a rota r_{du} não se altera implicando $\Delta cru=0$.
- $q_{djv}+q=0$ e j faz parte de r_{du}, então a entrega a j é retirada da rota r_{du} implicando $\Delta cru<0$.
- $q_{djv}+q>0$ e j não faz parte de r_{du}, então uma entrega a j é incluída na rota r_{du} implicando $\Delta cru>0$.
- $q_{djv}+q>0$ e j já faz parte de r_{du}, então a rota r_{du} não se altera implicando $\Delta cru=0$.

A linha 23 verifica se o valor da função objetivo devido às parcelas Δce, Δcrv e Δcru diminui e se a entrega q_{div} pode ser eliminada. Em caso afirmativo, o movimento é aceito e as variáveis são atualizadas. Essa implementação mostra uma forma sistemática de se tratar quaisquer número e tipos de restrições a partir do equacionamento da programação matemática. Neste algoritmo, as variáveis e restrições relativas ao roteamento foram relaxadas e tratadas à parte pelas listas r_{dv} e pelas funções calculaRemoção(), calculaMelhorInserção(), executaRemoção() e executaMelhorInserção().

4 Resultados computacionais

O algoritmo descrito neste trabalho foi codificado em C++. Todos os testes foram realizados em um computador com um processador Intel(R) Core(TM)2 Duo E7400 com 2.9GHz e 4GB de memória, rodando com Windows 7 Enterprise.

O resultado obtido foi comparado com método ALNS proposto por [Coelho(2012)]. Os testes do método ALNS foram realizados em um grid de computadores com processadores Dual Core AMD Opteron(TM) com 2.2 GHz, cada um com 12GB de memória, rodando com sistema operacional Linux. Segundo [Coelho(2012)] o tempo limite de execução foi limitado a 3600 segundos.

Para avaliar o desempenho do algoritmo, foi utilizado um subconjunto das instâncias propostas por [Coelho(2012)] para múltiplos veículos que foram adaptadas a partir das instâncias propostas por [Archetti et al.(2007), Archetti et al.(2012)] para veículo único. O subconjunto testado contém 20 instâncias variando de 5 a 50 clientes., cada uma com 5 replicações, totalizando 100 diferentes instâncias. Elas são descritas como *low* ou *high*, onde *low* e *high* estão relacionados a baixo e alto custo de estoque, respectivamente. O algoritmo proposto foi testado a partir destas instâncias para os casos de 2 e 3 veículos. O algoritmo foi executado 10 vezes para cada instância, com o valor de NS=21 e tempo máximo de execução de 600 segundos. É importante destacar que a execução foi interrompida por tempo em praticamente todas as instâncias.

Tabela 4 – Resultados - 3 períodos

		Número de Veículos									
		2					3				
Tipo	Dimensão	ALNS	Melhor	Média	Desvio(%)	GAP(%)	ALNS	Melhor	Média	Desvio(%)	GAP(%)
High	5	2494,65	2514,72	2526,12	0,45	0,8	2879,3	2879,28	2888,63	0,32	0
	10	4774,99	4795,58	4870,02	1,55	0,43	5276,69	5282,27	5383,75	1,92	0,11
	15	5768,6	5793,47	5871,98	1,36	0,43	6143,17	6176,28	6262,72	1,4	0,54
	20	7644,26	7729,5	7803,77	0,96	1,12	8130,39	8166,68	8279,78	1,38	0,45
	25	9395,88	9481,35	9623,29	1,5	0,91	9849,23	9941,27	10089,75	1,49	0,93
	30	11230,1	11329,3	11427,41	0,87	0,88	11590,26	11640,22	11805,02	1,42	0,43
	35	11765,62	11870,42	12011,09	1,19	0,89	12251,5	12312,62	12489,02	1,43	0,5
	40	12938,1	12969,04	13096,5	0,98	0,24	13374,34	13381,1	13496,17	0,86	0,05
	45	14325,6	14337,96	14386,64	0,34	0,09	14692,62	14723,22	14820,9	0,66	0,21
	50	15895,42	15966,08	16104,79	0,87	0,44	16488,44	16613,92	16765,65	0,91	0,76
	Média	9623,32	9678,74	9772,16	0,97	0,58	10067,59	10111,69	10228,14	1,15	0,44
Low	5	1572,28	1589,87	1637,35	2,99	1,12	1963,11	1963,11	1988,28	1,28	0
	10	2349,42	2370,94	2447,59	3,23	0,92	2850,57	2864,9	2924,25	2,07	0,5
	15	2536,32	2569	2636,95	2,64	1,29	2911,2	2979,6	3034,59	1,85	2,35
	20	3084,6	3160,75	3257,84	3,07	2,47	3563,21	3625,14	3723,54	2,71	1,74
	25	3373,87	3532,38	3626,42	2,66	4,7	3865,72	3953	4077,04	3,14	2,26
	30	3603,26	3640,35	3757,79	3,23	1,03	3985,43	3968,73	4047,8	1,99	**-0,42**
	35	3811,3	3835,8	3935,16	2,59	0,64	4292,74	4279,31	4391,09	2,61	**-0,31**
	40	4104,22	4139,22	4214,23	1,81	0,85	4451,24	4499,61	4604,58	2,33	1,09
	45	4324,41	4307	4388,25	1,89	**-0,4**	4681,85	4711,43	4778,58	1,43	0,63
	50	4841,91	4882,53	4996,98	2,34	0,84	5391,42	5472,35	5583,32	2,03	1,5
	Média	3360,16	3402,78	3489,86	2,56	1,27	3795,65	3831,72	3915,31	2,18	0,95

As tabelas 4 e 5 resumem os resultados médios obtidos para 3 e 6 períodos respectivamente. A coluna *Dimensão* representa o número de clientes da instância; as colunas *ALNS* mostram os valores com os quais o método proposto está sendo comparado e conforme divulgado em http://www.leandro-coelho.com/wp-content/uploads/2012/07/basic_small.txt e http://www.leandro-coelho.com/wp-content/-uploads/2012/07/basic_large.txt; as colunas *Melhor* e *Média* representam, respectivamente, o melhor resultado e a média das 10 execuções do algoritmo proposto neste trabalho; as colunas *Desvio* mostram a variação entre as coluna *Média* e *Melhor*; e as colunas $GAP = 100 \cdot \left(\frac{Melhor - ALNS}{ALNS}\right)$ mostram a variação entre o melhor resultado e o melhor resultado do *ALNS*. Para as instâncias em que o GAP está em negrito, a solução obtida foi melhor que aquela pelo *ALNS*.

A tabela 6 mostra os valores médios dos desvios e dos gaps dos resultados.

5 Conclusões

Este trabalho tratou do Problema de Estoque com Rotamento com Múltiplos Veículos (PERMV) através de um método híbrido que utiliza, de forma interativa, um algoritmo ILS adaptado e programação matemática inteira mista.

A aceitação de um movimento em uma vizinhança é facilitada pelo fato das quantidades em cada

Tabela 5 – Resultados - 6 períodos

		Número de Veículos									
		2					3				
Tipo	Dimensão	ALNS	Melhor	Média	Desvio(%)	GAP(%)	ALNS	Melhor	Média	Desvio(%)	GAP(%)
High	5	6147,73	6146,85	6192,23	0,74	**-0,01**	7206,68	7224,42	7318,89	1,31	0,25
	10	9803,98	9849,84	10028,84	1,82	0,47	11053,62	11075,9	11254,89	1,62	0,2
	15	12601,52	12665,62	12946,48	2,22	0,51	13814,68	13832,58	14023,37	1,38	0,13
	20	15934,08	16060,98	16340,25	1,74	0,8	17285,32	17423	17683,92	1,5	0,8
	25	18194,68	18495,26	18811,09	1,71	1,65	19573,78	19852,78	20171,12	1,6	1,43
	30	12536,4	12643,71	12863,78	1,74	0,86	13786,82	13881,74	14090,44	1,5	0,69
	Média	12536,4	12643,71	12863,78	1,74	0,86	13786,82	13881,74	14090,44	1,5	0,69
Low	5	3926,48	3921,09	3969,07	1,22	**-0,14**	4990,03	5039,17	5117,45	1,55	0,98
	10	5793,91	5875,11	6011,44	2,32	1,4	7177,63	7105,94	7227,14	1,71	**-1,00**
	15	6433,09	6545,89	6764,41	3,34	1,75	7607,58	7678,07	7883,29	2,67	0,93
	20	7875,37	8024,56	8212,78	2,35	1,89	9320,24	9402,17	9593,76	2,04	0,88
	25	8605,21	8720,09	9065,3	3,96	1,34	10234,46	10317,69	10511,61	1,88	0,81
	30	9054,79	9157,38	9434,26	3,02	1,13	10290,92	10283,01	10556,21	2,66	**-0,08**
	Média	6948,14	7040,69	7242,88	2,87	1,33	8270,14	8304,34	8481,58	2,13	0,41

Tabela 6 – Resumo dos resultados

	Número de Períodos							
	3				6			
	Número de Veículos							
	2		3		2		3	
	High	Low	High	Low	High	Low	High	Low
Desvio(%)	0,97	2,56	1,15	2,18	1,74	2,87	1,50	2,13
GAP(%)	0,58	1,27	0,44	0,95	0,86	1,33	0,69	0,41

entrega serem calculadas a partir dos equacionamentos das restrições, não sendo necessário definir-se heurísticas que aceitem ou rejeitem um movimento baseado nas quantidades atuais. Adicionalmente, os valores dessas entregas tem a tendência a ser os melhores valores possíveis considerados os clientes visitados em cada rota. Ou seja, uma vez que os clientes de cada rota estejam definidos, as quantidades entregues a eles já estão nos limites das restrições.

Os resultados obtidos foram bons quando comparados com o métodos ALNS levando-se em consideração que o tempos de execução foram bem inferiores. O critério de parada por tempo, atingido em quase todas as instâncias, indica que o algoritmo tem boa capacidade de diversificação mas precisa de mais intensificação nas buscas locais.

Em trabalhos futuros, pretende-se executar o algoritmo proposto para todo o conjunto de instâncias, principalmente aquelas com 50, 100 e 200 clientes, em que experimentos iniciais já mostraram que resultados significativamente melhores são alcançados quando comparado ao ALNS.

Referências

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

[Archetti et al.(2012)] **Archetti, C., Bertazzi, L., Laporte, G., and Speranza, M. G.** (2012). A hybrid heuristic for an inventory routing problem. *INFORMS Journal on Computing*, 24(1):101–116.

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

[Bertazzi et al.(2002)] **Bertazzi, L., Paletta, G., and Speranza, M. G.** (2002). Deterministic order-up-to level policies in an inventory routing problem. *Transportation Science*, 36(1):119–132.

[Campbell et al.(1998)] **Campbell, A., Clarke, l., Kleywegt, A. A., and Savelsbergh, M.** (1998). *The Inventory Routing Problem*, pages 95–113. Fleet Management and Logistics.

[Coelho(2012)] **Coelho, L. C.** (2012). *Flexibility and Consistency in Inventory-Routing*. PhD thesis, UNIVERSITÉ DE MONTRÉAL.

[Coelho et al.(2012)] **Coelho, L. C., Cordeau, J. F., and Laporte, G.** (2012). Consistency in multi-vehicle inventory-routing. *Transportation Research Part C*, 24:270–287.

[Coelho et al.(2013)] **Coelho, L. C., Cordeau, J. F., and Laporte, G.** (2013). Thirty years of inventory routing. *Transportation Science*, pages 1–19. ISSN 1526-5447.

[Gendreau and Potvin(2010)] Gendreau, M. and Potvin, J.-Y., editors (2010). *Handbook of Metaheuristics*, volume 146 of *International Series in Operations Research Management Science*. Springer.

[Penna(2013)] **Penna, P. H. V.** (2013). *Um algoritmo unificado para uma classe de problemas de roteamento de veículos com frota heterogênea*. PhD thesis, Universidade Federal Fluminense - UFF.

[Penna et al.(2013)a] **Penna, P. H. V., Subramanian, A., and Ochi, L. S.** (2013a). An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Journal of Heuristics*, 19(2):201–232. Print ISSN 1381-1231, Publisher Springer US.

[Penna et al.(2013)b] **Penna, P. H. V., Subramanian, A., Souza, M. J. F., and Ochi, L. S.** (2013b). *Uma heuristica eficiente para Problemas de Roteamento de Veículos*. Editora OMNIPAX. Capítulo de livro (Capítulo 25), pg: 165-180.

[Subramanian(2012)] **Subramanian, A.** (2012). *Heuristic, Exact and Hybrid Approaches for Vehicle Routing Problems*. tese de doutorado, Universidade Federal Fluminense.

[Subramanian et al.(2012)] **Subramanian, A., Penna, P. H. V., and Ochi, L. S.** (2012). A hybrid algorithm for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research - Elsevier*, 221:285–295.

[Subramanian et al.(2013)] **Subramanian, A., Uchoa, E., and Ochi, L. S.** (2013). A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, 40:2519–2531.