ESTRATÉGIAS *"CLUSTER FIRST AND THEN ROUTE"* PARA A RESOLUÇÃO DO PROBLEMA DO ROTEAMENTO DE VEÍCULOS CAPACITADOS: UM ESTUDO COMPARATIVO ENTRE TÉCNICAS DE ROTEIRIZAÇÃOS

Thiago André Guimarães
Centro Universitário Franciscano do Paraná
thiandgui@gmail.com

Luis Gustavo Pereira
Universidade Tecnológica Federal do Paraná
lgp1985@yahoo.com.br

Wesley José Nogueira Medeiros
Universidade Tecnológica Federal do Paraná
nogueira.wjnm@gmail.com

Lilian Caroline Xavier Candido
Programa de Pós-Graduação em Métodos Numéricos em Engenharia
Universidade Federal do Paraná
lilicarolinex@gmail.com

Resumo

O Problema do Roteamento de Veículos Capacitados (PRVC) é uma conhecida abordagem na resolução de problemas em logística de distribuição. Por sua complexidade combinatória, a solução do PRVC enseja o emprego de técnicas heurísticas. Neste sentido, este artigo apresenta e compara duas abordagens para a resolução do problema. O método emprega uma estratégia em dois estágios que primeiramente agrupa os clientes de acordo com a demanda e, posteriormente, constrói as rotas para os grupos formados. Na primeira abordagem, o agrupamento é feito por simulação de Monte Carlo, enquanto as rotas são geradas pela heurística do vizinho mais próximo e refinadas pelas trocas 2-opt. A segunda abordagem emprega a heurística de inserção mais econômica para a geração do roteiro inicial. Técnicas de computação paralela são utilizadas para a minimização do tempo de processamento. Experimentos computacionais realizados sobre instâncias clássicas da literatura apontam um bom desempenho de ambas as abordagens. Comparativamente, o desempenho das heurísticas propostas é equivalente, entretanto, a segunda abordagem demanda um tempo computacional superior.

Palavras-chave: roteamento de veículos capacitados; simulação de Monte Carlo; procedimentos heurísticos; agrupamento de pontos de demanda; paralelismo.

Abstract

The Capacitated Vehicle Routing Problem (PRVC) is a well-known approach to solving problems in distribution logistics. For its combinatorial complexity, the solution of PRVC motivates the use of heuristic techniques. In this sense, this paper presents and compares two approaches to solving the problem. The method employs a two-stage strategy that first grouping customers according to demand and then build the routes to the groups formed. In the first approach, the grouping is done by Monte Carlo simulation, while the routes are generated by the nearest neighbor heuristic and refined by the 2-opt procedure. The second approach uses the most economical insertion heuristic for generating the initial route. Parallel computer techniques are used to minimize the processing time. Computational experiments performed on instances of classical literature indicate a good performance of both approaches. Comparatively, the performance of heuristics is equivalent, however, the second approach requires a greater computational time.

Keywords: capacitated vehicle routing problem; Monte Carlo simulation, heuristic procedures; demand point clustering, parallel computing.

1. INTRODUÇÃO

O Problema do Roteamento de Veículos Capacitados (PRVC) vem sendo uma das mais importantes abordagens para a otimização em logística, desde que foi proposto inicialmente por Dantzig e Ramser (1959), decorrente da análise de um estudo sobre a distribuição de gasolina transportada por caminhões capacitados.

O PRVC busca determinar um conjunto de rotas para uma frota homogênea de veículos, partindo de um depósito central com destino a um conjunto de clientes dispersos geograficamente e que demandam determinado produto. Cada cliente deve ser atendido por apenas um veículo (embora ocorram variações no problema original que permitem entregas fracionadas) e, além disso, a demanda transportada por cada um dos veículos da frota não deve exceder sua capacidade de carregamento. O objetivo do PRVC é minimizar a distância total percorrida pela frota de veículos.

Entre as diversas abordagens para a resolução do PRVC, pode-se classificá-las em duas classes distintas: métodos exatos e métodos heurísticos. Referente à primeira classe, alguns dos trabalhos são baseados em algoritmos *branch-and-cut* ou relaxação lagrangeana/geração de colunas. Um dos marcos na abordagem exata para o PRVC foi apresentado em Christofides *et al.* (1981), que trabalharam com limitantes lagrangeanos para a geração de subrotas. O algoritmo branch-and-bound foi capaz de resolver instâncias com até 25 clientes. Já, Fukasawa *et al.* (2006) desenvolveram um algoritmo *branch-and-cut-and-price* sob instâncias com mais de 135 clientes, obtendo para todas elas resultados ótimos. Entretanto, problemas de grande porte ensejam um elevado esforço computacional para sua resolução, dado que o tempo de processamento aumenta exponencialmente com o incremento do número de clientes (nós). Devido a isto, métodos heurísticos vêm sendo propostos para a resolução do PRVC.

Neste contexto, com relação à segunda classe (métodos heurísticos), Campos e Mota (2000) apresentaram duas heurísticas: uma baseada em *scratch*, que gera uma solução inicial sem qualquer informação obtida *a priori*, e outra que emprega informações advindas de relaxações lineares fortes a partir do problema original. Os autores utilizaram técnicas de busca tabu para refinar as soluções iniciais. As heurísticas foram testadas sob instâncias da literatura com a quantidade de clientes variando de 22 a135 clientes, obtendo soluções ótimas para a maioria delas. Por sua vez, Chen *et al.* (2006) propuseram uma abordagem híbrida baseada em nuvem de partículas e testaram sob instâncias clássicas com uma faixa de clientes variando de 33 a 155. Os resultados obtidos se aproximaram dos valores ótimos.

Comumente é reportada na literatura uma estratégia em duas fases para a resolução do PRVC. Esta estratégia primeiramente agrupa os clientes conforme a localização e quantidade demandada para posteriormente construir rotas para os grupos formados (*cluster first and then route*). Garantido que a demanda dos clientes pertencentes a um grupo não exceda a capacidade do veículo, o PRVC recai na resolução do Problema do Caixeiro Viajante (PCV), ensejando um roteiro para cada grupo, que passe por todos os pontos, e apresente a menor distância possível. Fisher e Jaikumar (1981) e Gillett e Johnson (1976) endereçam heurísticas do tipo "*cluster first and then route*" para a resolução do PRVC. Variações do problema clássico tratadas com essa mesma abordagem encontram-se em Sariklis and Powell (2000), que trabalham para a resolução do OVRP (*Open Vehicle Routing Problem*). O OVRP se diferencia do PRVC pela não necessidade de retorno do veículo ao depósito após visitar o último cliente da rota. O fato de a estratégia possibilitar o emprego de diferentes técnicas para cada uma das etapas é pertinente avaliar o efeito deste emprego diferenciado sobre a solução final.

Neste sentido, o presente estudo propõe e compara duas heurísticas do tipo "*cluster first and then route*" para resolução do PRVC. Em ambas, o agrupamento dos pontos de demanda é realizado através de técnicas de simulação de Monte Carlo, em conjunto com a heurística de designação de Gillet e Johnon (1976). Já para a fase de roteirização, a primeira abordagem emprega a heurística do vizinho mais próximo, enquanto a segunda utiliza a heurística de inserção mais econômica para a construção da solução inicial. O refinamento é obtido pelas trocas 2-*opt* e isso se aplica em ambas as abordagens. Técnicas de processamento paralelo são introduzidas em ambos os casos para reduzir o tempo de processamento. Dessa forma é possível analisar e quantificar o efeito da fase de roteirização sobre a solução final do problema.

Para tanto, o estudo se inicia com um referencial teórico, apresentado no item 2, onde o PRVC é formalmente definido. No item 3 discutem-se os principais algoritmos necessários à construção das estratégias propostas. Neste mesmo tópico as duas heurísticas são apresentadas. O item 4 apresenta e discute os resultados obtidos enquanto o item 5 tece as conclusões do artigo.

2. REFERENCIAL TEÓRICO

Esta seção introduz formalmente o PRVC e comenta sobre a técnica de processamento paralelo empregada na minimização dos tempos computacionais.

2.1. PROBLEMA DO ROTEAMENTO DE VEÍCULOS CAPACITADOS

O PRVC pode ser formalmente definido sob o seguinte aspecto: seja $G(V, E)$ um grafo não direcionado contendo o conjunto de vértices $V = \{0, 1, \dots, n\}$, onde o vértice "0" representa o depósito, enquanto que todos os outros vértices representam os clientes, sendo que cada cliente i possui uma demanda d_i. Cada arco $e \in E$ possui um comprimento não negativo $l(e)$. Dado G e dois números positivos e inteiros $(K\ e\ C)$, o PRVC consiste em encontrar um conjunto de rotas para os K veículos, que atenda as restrições:

(i) Cada rota inicia e termina no depósito;
(ii) Cada cliente é visitado por um único veículo;
(iii) A demanda total de todos os clientes de uma sub-rota não excede a capacidade C do veículo.

Uma instância exemplo é apresentada na Figura 1 a seguir. Dentro de cada nó está indicada a demanda referente ao próprio nó.

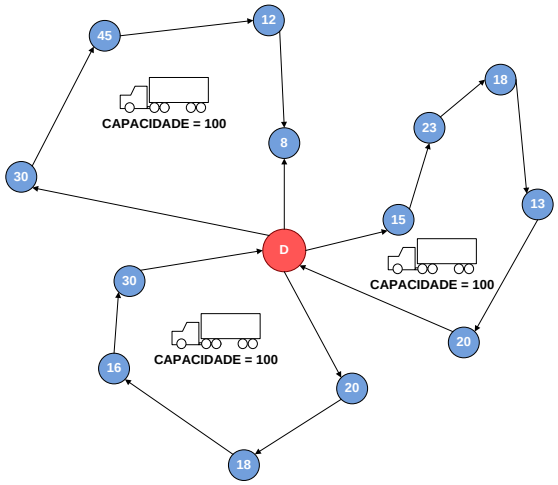

Figura 1 - Instância exemplo do PRVC Fonte: Elaborado pelos autores

O objetivo do PRVC é minimizar o somatório das distâncias de todas as sub-rotas. O problema é fortemente *NP-hard*, dado que é uma generalização do Problema do Caixeiro Viajante (PCV) demandando, portanto, elevado esforço computacional para obtenção da solução ótima através de busca exaustiva. Baseado em Lin *et al.* (2009), o PRVC pode ser modelado como um problema de programação inteira misto como segue:

Minimizar $$\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{k=1}^{K} C_{ij}X_{ijk} \quad (1)$$

Sujeito à: $$\sum_{i=0}^{N}\sum_{i=0}^{N} X_{ijk}d_i \leq Q^k \qquad 1 \leq k \leq K, \quad (2)$$

$$\sum_{j=1}^{N} X_{ijk} = \sum_{j=1}^{N} X_{ijk} \leq 1, para\ i = 0\ e\ k \in \{1, \dots, k\}, \quad (3)$$

$$\sum_{k=1}^{K}\sum_{j=1}^{N} X_{ijk} \leq K, para\ i = 0 \quad (4)$$

Onde C_{ij} é o custo para ir do cliente i para o cliente j; K é a quantidade de veículos disponível; N é o número de clientes; Q^k é a capacidade de carregamento do veículo k e finalmente d_i é a demanda do cliente i. As variáveis $X_{ijk} \in 0\ ou\ 1\ (i \neq j; i, j \in 0, 1, \dots, N)$. A equação (1) é a função objetivo que minimiza o somatório das distâncias de todas as sub-rotas. As restrições (2)

referem-se à capacidade de carregamento do veículo e atendimento das demandas de cada cliente, onde $X_{ijk} = 1$, se o veículo k viaja do cliente i para o cliente j diretamente e $X_{ijk} = 0$, caso contrário. As restrições (3) garantem que toda rota tem início e fim no depósito, enquanto que a restrição (4) especifica que há no máximo K rotas partindo do depósito.

2.2. PROCESSAMENTO PARALELO

Leijen e Hall (2007) indica que os computadores de vários núcleos tornaram-se o padrão industrial aplicado atualmente. Infelizmente, segundo os mesmo autores, a maioria dos aplicativos utiliza um só núcleo e não apresenta aumento de velocidade quando executados em uma máquina com vários núcleos. Dessa forma, para aprimorar o desempenho de um *software* torna-se necessário executá-lo em vários processadores em paralelo.

Novas ferramentas de programação foram desenvolvidas para empregar os recursos de paralelismo em computadores multinucleados. Dentre elas, Leijen e Hall (2007) destaca a classe *Parallel* presente no *.NET Framework 4* e exemplificada na Figura 1 a seguir:

```
System.Threading.Tasks.Parallel.For((0, 100, delegate (int i) {
    a[i] = a[i]*a[i];
});
```

Figura 2 - Exemplo de trecho paralelizado Fonte: Leijen e Hall (2007)

O presente estudo emprega a linguagem de programação C# 4.0 em conjunto com a extensão de linguagem denominada *Language Integrated Query* (LINQ), além da classe *Parallel*. O LINQ possibilita uma manipulação de conjuntos de dados e o a classe *Parallel* permite o devido processamento paralelo pelo software.

Embora represente um grande passo à frente, Toub (2011) reforça que a classe Parallel só chega a abordar superficialmente as funcionalidades disponíveis para paralelismo. Conforme o mesmo autor, um dos mais formidáveis avanços em paralelização foi o realizado pelo *.NET Framework 4* através da introdução da extensão *Parallel LINQ* (PLINQ). Essa extensão permite o processamento paralelo dos dados internos de um conjunto, desde que os mesmo possuam alto grau de independência linear. A Figura 3 exemplifica a utilização da extensão PLINQ – *AsParallel()* – em um trecho do código fonte desenvolvido neste trabalho:

```
var medianaProxima = (from m in mMedianas.AsParallel()
                      orderby m % designando.cliente.Coordenada
                      where m.CapacidadeDisponivel >=
designando.cliente.Demanda
                      select m).First();
```

Figura 3 - Exemplo de aplicação de PLINQ no software desenvolvido. Fonte: Elaborado pelo autor

3. ESTRATÉGIAS DE RESOLUÇÃO

Problemas de grande porte como o PRVC desmotivam o emprego de modelos exatos de programação linear dado que o tempo de processamento aumenta exponencialmente com o incremento do número de clientes (nós). Devido a isto, métodos heurísticos vêm sendo propostos para a resolução do PRVC no intuito de se produzir bons resultados com tempo de processamento factíveis.

A construção das heurísticas propostas neste artigo requer a resolução dos problemas de agrupamento de pontos de demanda e posteriormente a roteirização para cada grupo formado. Assim, as subseções a seguir discutem esses problemas de forma pormenorizada. Por fim, as abordagens são apresentadas.

3.1. AGRUPAMENTO DOS PONTOS DE DEMANDA

O problema das *p*-medianas é introduzido com o objetivo de determinar, no grafo formado pelos clientes, *p*-pontos principais que representem sementes ou depósitos fictícios. A cada uma destas medianas, os demais pontos de demanda serão designados, de modo a formar *p*-conjuntos iniciais (*p*-grupos) de pontos, cada um com "capacidade" igual à capacidade de carregamento dos veículos, que é a mesma para todos eles (frota homogênea). A estratégia de resolução reside na teoria dos grafos que tem por objetivo localizar facilidades (instalações) ao longo de uma rede viária. Estas facilidades (*p*-medianas) devem ser escolhidas de forma a minimizar a soma da distância de cada um dos pontos à

facilidade mais próxima, ponderada por um fator de demanda. Uma descrição mais formalizada para o problema é apresentada a seguir.

Para um grafo $G(V,E)$, define-se número de *out*-transmissão e *in*-transmissão, para cada ponto $v_i \in V$, como sendo, respectivamente:

$$\sigma_0(v_i) = \sum_{v_j \in V} \theta_j \cdot \omega(v_i, v_j)$$

$$\sigma_t(v_i) = \sum_{v_j \in V} \theta_j \cdot \omega(v_i, v_j)$$

onde, o vetor $\omega(v_i, v_j)$ é a distância entre o vértice v_i e v_j, e θ_j é o peso associado ao ponto v_j. De acordo com Christofides (1975), são chamadas de *out*-mediana e *in*-mediana de um grafo, respectivamente, os pontos $\overline{v_i}$ e $\overline{v_j}$ que satisfazem as seguintes condições:

$$\sigma_0(\bar{v}_0) = \min_{v_i \in V}[\sigma_0(v_i)]$$

$$\sigma_t(\bar{v}_t) = \min_{v_i \in V}[\sigma_t(v_i)]$$

Para generalizar estes conceitos de *out*-transmissão e *in*-transmissão para *p*-medianas, considera-se V_p um subconjunto do conjunto de pontos V do grafo $G(V,E)$, que possui p elementos, isto é, a cardinalidade de V_p é p (ou, simplesmente, $|V_p| = p$). Define-se ainda, $\omega(V_p, v_j) = \min_{v_i \in V}[\omega(v_i, v_j)]$ e $\omega(v_j, V_p) = \min_{v_i \in V}[\omega(v_j, v_i)]$, onde $\omega(V_p, v_j)$ representa a distância do subconjunto de pontos V_p até o ponto v_j e $\omega(v_j, V_p)$ indica a distância do vértice v_j até o subconjunto V_p. Analogamente ao procedimento para um só vértice, definem-se os números de out-transmissão e in-transmissão, respectivamente, para o conjunto V_p, da seguinte forma:

$$\sigma_0(V_p) = \sum_{v_j \in V} \theta_j \cdot \omega(V_p, v_j)$$

$$\sigma_t(V_p) = \sum_{v_j \in V} \theta_j \cdot \omega(v_j, V_p)$$

Finalmente, denomina-se de *p*-*out*-mediana e *p*-*in*-mediana os conjuntos $\bar{V}_{po}$ e $\bar{V}_{pt}$, respectivamente, para os quais:

$$\sigma_0(\bar{V}_{po}) = \min_{V_p \subset V}[\sigma_0(V_p)]$$

$$\sigma_t(\bar{V}_{pt}) = \min_{V_p \subset V}[\sigma_t(V_p)]$$

Neste trabalho, considera-se um grafo não orientado, tornando indiferente o conceito de *p*-*in*-mediana ou *p*-*out*-mediano, que doravante será denominado apenas de *p*-mediana, simplificando consideravelmente a notação utilizada.

O problema das *p*-medianas pode ser resolvido de forma exata (HAKIMI, 1965), usando enumeração exaustiva ou programação inteira. Todavia, para problemas de grande porte (grande quantidade de pontos), torna-se necessário recorrer a métodos aproximados (heurísticos) tendo em vista o esforço computacional requerido pelos métodos exatos ser muito elevado.

3.1.1. RESOLUÇÃO DO PROBLEMA DAS *P*-MEDIANAS VIA SIMULAÇÃO DE MONTE CARLO

Segundo Juan *et al.* (2010), recentes avanços no desenvolvimento de geradores números pseudoaleatórios vem abrindo novas perspectivas para o uso de simulação de Monte Carlo em problemas de otimização. Com este avanço, a estratégia aplicando o método estocástico de Monte Carlo para resolução do problema das *p*-medianas, para posterior designação dos pontos de demanda às medianas geradas se apresenta viável.

Para a determinação destes *p*-pontos iniciais, foi utilizado o gerador de números pseudoaleatórios existente no *.NET Framework 4*. Essa ferramenta garante que a escolha dos pontos iniciais é feita de maneira estocástica com distribuição normal. A orientação citada baseia-se na delimitação de uma a área para a atuação da simulação de Monte Carlo. Essa área foi criada através de um retângulo definido pelas mínimas e máximas coordenadas dos pontos do grafo, em seus respectivos eixos de abscissas e ordenadas. Dessa forma, para cada *p*-ponto do problema gera-se dois números pseudoaleatórios, sendo um para a coordenada x e outro para a coordenada y do ponto. Os números pseudoaleatórios pertencem ao intervalo entre a mínima e a máxima abcissa e a mínima e

máxima ordenada definida anteriormente. Tal delimitação visa otimizar o processo de geração de pontos pelo método de Monte Carlo, uma vez que elimina a possibilidade de serem escolhidos pontos demasiadamente distantes e externos ao grafo do PRVC. A Figura 4 apresenta o algoritmo.

```
    Procedimento MonteCarlo (destinos, quantidadeVeiculos, capacidadeVeiculo)
1   Início
2       xMin ← Min destinos.X
3       xMax ← Max destinos.X
4       yMin ← Min destinos.Y
5       yMax ← Max destinos.Y
6       Para iteração ← 1 até quantidadeVeiculos faça
7           RetornoIterado nova mediana(X ← aleatório * (xMax - xMin) + xMin,
8           Y ← aleatório * (yMax - yMin) + yMin,
9           capacidade ← capacidadeVeiculo)
10      FimPara
11  Fim
```

Figura 4 - Implementação do Monte Carlo, Fonte: Elaborado pelos Autores

3.1.2.RESOLUÇÃO DO PROBLEMA DAS *P*-MEDIANAS VIA GILLET E JOHNSON (1976)

Uma vez obtido o conjunto de medianas S, faz-se necessário designar de maneira ótima (ou quase ótima) os pontos pertencentes ao conjunto $(V - S)$ a elas formando. Desta maneira, os agrupamentos serão atendidos pela frota disponível de veículos. A designação é realizada respeitando a capacidade da mediana (determinada pela capacidade de carregamento do veículo). Para a designação dos pontos, Gillet e Johnson (1976) propuseram um procedimento heurístico conhecido como algoritmo de Gillet e Johnson. A ideia básica do referido algoritmo é selecionar os pontos que apresentam a maior razão entre as distâncias às duas medianas mais próximas para serem designados com prioridade. Este procedimento prioriza a designação de pontos que, senão forem designados às medianas mais próximas, terão uma penalização maior. A Figura 5 apresenta o algoritmo.

```
    Procedimento GilletJohnson (destinos, demandaPonto, quantidadeVeiculos, capacidadeVeiculo)
1   Início
2   t_i^1 ← PrimeiraMediana mais próxima ao ponto i
    t_i^2 ← SegundaMediana mais próxima ao ponto i
    c_i^1 ←Distância da PrimeiraMediana mais próxima ao ponto i
    c_i^2 ← Distância da SegundaMediana mais próxima ao ponto i
3   Para ponto ← 1 até destinos faça
4       Obter t_i^1 e t_i^2
5       Calcular c_i^1 e c_i^2
6       Calcular a razão r_i = c_i^1 / c_i^2
7   FimPara
8   Ordenar os pontos de acordo com r_i em ordem decrescente
9   Para mediana← 1 até quantidadeVeiculos faça
10      Enquanto demandaPonto < capacidadeVeiculoponto ← 1 até destinos faça
11          Designar ponto para mediana
12      FimEnquanto
13  FimPara
14  Para ponto ← 1 até destinos faça
15      Se ∃ ponto não designado
16          Volte ao passo 3
17      FimSe
18  FimPara
19  Fim
```

Figura 5 - Algoritmo de Gillet e Johnson - Fonte: Gillet e Johnson (1976)

Concluído o agrupamento dos pontos de demanda, o Problema do Roteamento de Veículos Capacitados recai no Problema do Caixeiro Viajante. Doravante, será explicitado as técnicas para a construção e a melhoria das rotas.

3.2. PROBLEMA DA ROTEIRIZAÇÃO: GERAÇÃO DA SOLUÇÃO INICIAL

A resolução do problema do roteamento enseja a obtenção de uma solução inicial para posterior refinamento. Para a abordagem proposta neste trabalho foi utilizado o algoritmo do Vizinho Mais Próximo (VMP) pela sua ampla divulgação na literatura existente e fácil implementação computacional.

O algoritmo de roteirização VMP foi originalmente proposto por Cover e Hart (1967) e consiste na composição da rota com base na inserção sequencial de pontos através de um ponto inicial conforme a menor distância entre este e os seus demais pontos (vizinhos). Após determinar todas as distâncias entre o ponto inicial e os demais pontos do cluster, os demais pontos são ordenados de maneira decrescente para possibilitar a escolha e determinação do nó mais próximo que será designado à rota. O algoritmo está exposto na Figura 6 a seguir:

	Procedimento VizinhoMaisProximo (*destinos*, *agrupamentos*)
1	Início
2	Iniciar a roteirização a partir do depósito e designar este ponto como ponto i visitado ($i \in S$), onde S é o grupo de clientes visitados;
3	Para cada ponto ($j \notin S$), obter d_{ij} sendo este a respectiva distância entre o ponto i e j.
4	Escolher o menor d_{ij} e marcar o ponto j como visitado;
5	Faça o ponto j ser o ponto i;
6	**Se** todos os pontos já foram visitados **faça**
7	Fim
8	Senão
9	Volte ao passo 2
10	FimSe
11	Fim

Figura 6 - Algoritmo VMP. Fonte: Cover e Hart (1967)

A segunda abordagem proposta emprega a heurística de inserção mais econômica para a geração do roteiro inicial. Este método constrói uma rota inicial e factível, atendendo a um conjunto de critérios que garantam certa eficiência para o refinamento da solução obtida. Uma descrição mais formal da heurística é apresentada por Steiner et al. (2000), cuja reprodução encontra-se na figura 7 a seguir.

	Procedimento InsercaoMaisEconomica (*destinos*, *agrupamentos)*
1	**Início**
2	p ←agrupamentos
3	Rota←∅
3	Comece com um subgrafo consistindo somente do nó $p;$
4	Encontre o nó m tal que $dist(p,m)$ seja mínima
5	Rota←p-m-p
6	**Enquanto** ∃ *destinos* ∉ *Rota* , encontre k tal que $dist\,(p,k) + dist\,(k,m) - dist\,(p,m) = mínimo$
8	**Se** $dist(p,k,m,p) < dist(p,m,k,p)$ **então**
9	Rota←p-k-m-p
10	**Senão**
11	Rota←p-m-k-p
12	**FimSe**
13	**FimEnquanto**
19	**Fim**

Figura 7 - Heurística de Inserção Mais Econômica - Fonte: Steiner et al. (2000)

3.3. PROBLEMA DA ROTEIRIZAÇÃO: REFINAMENTO DA SOLUÇÃO INICIAL

O problema da melhoria de rotas busca aumentar a eficiência do trajeto, ou seja, refinar uma solução inicial obtida para uma rota a ser seguida, a priori, por um veículo. Uma das melhores abordagens heurísticas para a resolução desse problema é a proposta por Lin e Kernigham (1973) denominada troca de arcos *k-opt*, onde as trocas *2-opt* (2 arcos) e *3-opt* (3 arcos) são as mais utilizadas.

Os métodos *k-opt* buscam a melhoria de um trajeto pela substituição de "k" arcos no roteiro estabelecido anteriormente, isto é, "k" arcos são removidos do roteiro e substituídos por outros "k" arcos. Caso alguma melhoria seja detectada, a troca é aceita e o novo arco passa a compor a solução incumbente. Neste artigo, esta dinâmica se repete até que nenhuma troca outra troca resulte em melhoria.

Conforme apontado por Laporte (1999), o processo de melhoria *k-opt* termina em um mínimo local e possui ordem de complexidade *O(nk)*. Quanto maior for o valor de *k*, melhor será a solução, entretanto o esforço computacional requerido também será maior. Isto leva a um trade-off entre qualidade e tempo computacional. Dessa forma, trocas 4-opt e superiores ensejam um custo computacional muitas vezes superior à melhoria da solução obtida.

Aqui optou-se por utilizar a troca *2-opt* pela sua simplicidade de implementação, uma vez que esta estratégia de melhoria realiza a comparação somente entre dois trechos para então determinar se a inversão de sentido entre eles é mais econômica em relação à rota inicial. Caso seja, a nova rota passa a ser a rota preferencial.

Na Figura 7 abaixo, é ilustrada a sistemática de troca de arcos para a abordagem 2-opt que será utilizada na heurística proposta no trabalho:

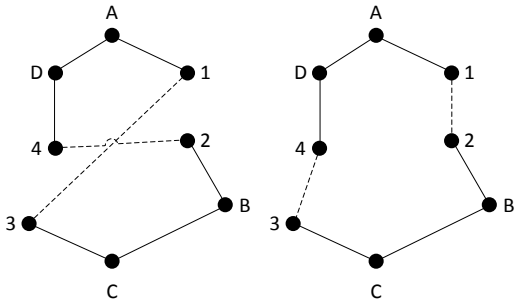

Figura 8 - Troca 2-opt entre os trechos 1-2 e 3-4 Fonte: Adptado de Costa (1997)

Cabe destacar que a geração de boas soluções iniciais é fundamental para que se obtenham boas soluções finais com os métodos k-opt para a melhoria de rotas (COSTA, 1997).

3.4. HEURÍSTICAS PROPOSTAS

As duas heurísticas propostas iniciam-se com o agrupamento dos clientes via simulação de Monte Carlo, ajustado *a priori*, pela definição da quantidade de medianas conforme a quantidade de veículos disponíveis. A designação dos pontos às medianas é feito pelo algoritmo de Gillet e Johnson (1976), formando os agrupamentos de clientes para serem roteirizados. Na sequência cada agrupamento é roteirizado, sendo a primeira rota inicial é gerada pela heurística construtiva do vizinho mais próximo e a segunda abordagem pela heurística de inserção mais econômica. Para os dois casos, a solução inicial é refinada pela heurística de melhoria 2-opt. Denomina-se, portanto, a primeira abordagem como SMC&GJ+VMP+2-*opt*, enquanto a segunda abordagem de SMC&GJ+IME+2-*opt*. A Figura 9 e Figura 10 detalham os algoritmos.

```
     Procedimento SMC&GJ+VMP+2-opt (instância)
1    Início
2        iteração ← 0
3        Enquanto iteração < Número Máximo faz
4            SoluçãoInicial ← ProcessarMonteCarloComGilletJohnson(instância)
5            Roteirização ← ProcessarVizinhoMaisProximo(SoluçãoInicial,
     instância)
6            Resposta ← 2-Opt(Roteirização, instância)
7            iteração ← iteração + 1
8            Se Resposta.CustoTotal < Solução.CustoTotal então
9                Solução ← Resposta
10           FimSe
11       FimEnquanto
12       Retorna Solução
13   Fim
```

Figura 9 - Abordagem 1 – Vizinho Mais Próximo. Fonte: Elaborado pelos Autores

```
     Procedimento SMC&GJ+IME+2-opt (instância)
1    Início
2        iteração ← 0
3        Enquanto iteração < Número Máximo faz
4            SoluçãoInicial ← ProcessarMonteCarloComGilletJohnson(instância)
5            Roteirização ← ProcessarInserçãoMaisEconomica(SoluçãoInicial,
     instância)
6            Resposta ← 2-Opt(Roteirização, instância)
7            iteração ← iteração + 1
8            Se Resposta.CustoTotal < Solução.CustoTotal então
9                Solução ← Resposta
10           FimSe
11       FimEnquanto
12       Retorna Solução
13   Fim
```

Figura 10 - Abordagem 2 – Inserção Mais Econômica. Fonte: Elaborado pelos Autores

4. RESULTADOS OBTIDOS

Os experimentos computacionais foram realizados sobre as instâncias clássicas da literatura, disponíveis no sítio *http://www.branchandcut.org*. Foram selecionadas 35 instâncias de 3 classes distintas. A diferença existente entre as classes de instâncias pode variar desde a região onde os clientes estão distribuídos até a densidade dos mesmos em relação à área do plano. Quanto ao hardware utilizado, as heurísticas foram executadas em um processador Intel® Core™ 2 Quad Q6600, 2.4 GHz com 4 GB de RAM e sistema operacional Windows 7™, 64-Bits. Com relação aos critérios de parada, utilizou-se 1000 simulações para cada instância.

A tabela 1, apresentada a seguir, contém os resultados para o conjunto de 35 instâncias testadas. Na primeira coluna está o nome da instância. Já na segunda (BKS – *Best Known Solution*) apresenta-se a melhor solução conhecida (solução ótima). As três próximas apresentam para abordagem 1 (SMC&GJ+VMP+2-*opt*) qual a melhor solução encontrada dentre todas as iterações, o desvio percentual em relação à BKS e o tempo de processamento para 1000 simulações para todas as instâncias. As outras três colunas contém as mesmas informações para a abordagem 2 (SMC&GJ+IME+2-*opt*). As duas últimas colunas comparam a solução entre ambas as abordagens e a diferença de tempo de processamento.

Tabela 1 - Síntese dos Resultados Obtidos. Fonte: Elaborado pelos Autores

Instância	BKS	SMC-VMP-2Opt	Desvio BKS	Tempo (s)	SMC-IME-2Opt	Desvio (%)	Tempo (s)	GAP VMP/IME	GAP Tempo
A-n32-k5	784	798	1,8%	2,4	792	1,0%	3,0	-0,8%	-20,3%
A-n34-k5	778	790	1,5%	2,3	790	1,5%	2,0	0,0%	13,0%
A-n36-k5	799	825	3,3%	2,6	816	2,1%	2,0	-1,1%	28,0%
A-n37-k6	949	989	4,2%	2,3	984	3,7%	3,0	-0,5%	-25,0%
A-n38-k5	730	755	3,4%	2,5	749	2,6%	4,0	-0,8%	-37,0%
A-n39-k6	831	868	4,5%	2,5	869	4,6%	3,0	0,1%	-15,3%
A-n44-k6	937	984	5,0%	3,0	965	3,0%	3,0	-2,0%	1,4%
A-n48-k7	1073	1133	5,6%	3,6	1125	4,8%	4,0	-0,7%	-10,3%
A-n53-k7	1010	1088	7,7%	4,0	1104	9,3%	5,0	1,4%	-20,2%
A-n54-k7	1167	1261	8,1%	4,3	1227	5,1%	5,0	-2,8%	-14,6%
A-n60-k9	1354	1439	6,3%	4,4	1434	5,9%	4,0	-0,3%	9,0%
A-n62-k8	1288	1403	8,9%	5,0	1412	9,6%	6,0	0,6%	-17,3%
A-n63-k9	1616	1753	8,5%	3,9	1774	9,8%	6,0	1,2%	-35,5%
A-n64-k9	1401	1481	5,7%	4,8	1550	10,6%	6,0	4,5%	-19,3%
A-n80-k10	1763	1920	8,9%	6,6	1961	11,2%	9,0	2,1%	-26,7%

B-n31-k5	672	683	1,6%	2,0	680	1,2%	2,0	-0,4%	-2,0%
B-n38-k6	805	828	2,9%	2,5	827	2,7%	3,0	-0,1%	-18,0%
B-n39-k5	549	563	2,6%	3,1	564	2,7%	3,0	0,2%	3,4%
B-n43-k6	742	765	3,1%	3,1	757	2,0%	4,0	-1,1%	-22,3%
B-n44-k7	909	942	3,6%	3,0	947	4,2%	5,0	0,5%	-40,6%
B-n45-k6	678	695	2,5%	2,6	722	6,5%	4,0	3,7%	-36,0%
B-n50-k8	1312	1361	3,7%	3,9	1357	3,4%	5,0	-0,3%	-23,0%
B-n52-k7	747	780	4,4%	4,0	769	2,9%	4,0	-1,4%	0,0%
B-n56-k7	707	755	6,8%	5,0	753	6,5%	6,0	-0,3%	-17,2%
B-n57-k9	1598	1636	2,4%	3,9	1631	2,1%	5,0	-0,3%	-21,2%
B-n68-k9	1272	1319	3,7%	5,5	1336	5,0%	4,0	1,3%	38,3%
B-n78-k10	1221	1350	10,6%	6,5	1342	9,9%	9,0	-0,6%	-28,0%
E-n22-k4	375	375	0,0%	1,2	375	0,0%	2,0	0,0%	-38,0%
E-n23-k3	569	569	0,0%	1,6	569	0,0%	2,0	0,0%	-18,5%
E-n30-k3	534	536	0,4%	2,2	536	0,4%	3,0	0,0%	-25,3%
E-n33-k4	835	852	2,0%	2,6	845	1,2%	2,0	-0,8%	28,5%
E-n51-k5	521	548	5,2%	4,9	540	3,6%	5,0	-1,5%	-2,4%
E-n76-k7	682	756	10,9%	9,1	740	8,5%	9,0	-2,2%	0,6%
E-n76-k8	735	814	10,7%	7,4	825	12,2%	8,0	1,3%	-7,0%
E-n101-k8	817	906	10,9%	14,3	890	8,9%	14,0	-1,8%	2,4%
Média			4,9%	4,1		4,8%	4,7	-0,1%	-11,9%

Em termos médios, o desempenho da heurística SMC&GJ+IME+2-opt, foi ligeiramente superior à heurística SMC&GJ+VMP+2-opt, com um desvio 0,1 ponto percentual acima. Para ambos os casos, os resultados obtidos demonstram-se bastante satisfatórios. Já com relação ao tempo de processamento, verifica-se a eficiência das técnicas de paralelismo, pois o tempo médio para 1000 simulações ficou pouco acima de 4,1 segundos na primeira abordagem e não chegou a atingir 4,7 segundos na segunda. Comparativamente, o desempenho médio do tempo de processamento demonstrou que a utilização da heurística do vizinho mais próximo é praticamente 12% mais rápida que a heurística de inserção mais econômica.

5. CONCLUSÕES

O presente artigo apresentou duas propostas para a resolução do Problema do Roteamento de Veículos Capacitados, com base na estratégia "*Cluster First and then Route*", que consiste primeiramente em agrupar os pontos de demanda para posteriormente construir as rotas para os grupos formados. Para a primeira proposta, empregou-se simulação de monte carlo em conjunto com o algoritmo de Gillet Johnson (1976) para a resolução do problema do agrupamento. A roteirização dos pontos foi feita através a heurística do vizinho mais próximo, com refinamento 2-opt. A segunda abordagem diferencia-se da primeira por empregar a heurística de inserção mais econômica para a geração do roteiro inicial. Em ambos os casos, técnicas de processamento paralelo foram utilizadas para reduzir o tempo de processamento.

Os resultados obtidos demonstram que ambas as abordagens são eficientes para a resolução do problema, com desvios médios não superiores à 5% em ambos os casos, com reduzido tempo de processamento. Todavia, a análise comparativa entre elas aponta um desempenho ligeiramente superior da abordagem 2 sobre a abordagem 1, demonstrando que a heurística de inserção mais econômica possibilita uma melhor solução global para o PRVC. Entretanto, essa ligeira vantagem implica em um tempo médio de processamento aproximadamente 12% maior.

6. REFERÊNCIAS BIBLIOGRÁFICAS

[1] CAMPOS, V.; MOTA, E. Heuristic Procedures for the Capacitated Vehicle Routing Problem, **Computational Optimization and Applications**, v. 16, p. 265 – 277, 2000.

[2] CHEN A.; YANG G.; WU Z. Hybrid discrete particle swarm optimization algorithm for capacitated vehicle routing problem. **Journal of Zhejiang University Science**, v. 7, p. 607–614, 2006.

[3] CHRISTOFIDES, N., **Graph Theory: An algorithmic approach**. New York: Academic Press Inc,1975.

[4] COSTA, D. M. B. **Aplicação de Algumas técnicas da Pesquisa Operacional na Otimização de Serviços Postais. Dissertação** (Mestrado em Métodos Numéricos em Engenharia), Universidade Federal do Paraná, Curitiba, 1997.

[5] COVER, T.; HART, P.; Nearest neighbor pattern classification. Stanford: Information Theory, **IEEE Transactions on Operational Research**, v.13, n.1, p. 21- 27, 1967.

[6] DANTZIG, G.; RAMSER, R. The truck dispatching problem. **Management Science**, v. 6, p. 80–91, 1959.

[7] FISHER, M. L.; JAIKUMAR, R. A generalized assignment heuristic for vehicle routing. **Networks**,v.11, p.109–124, 1981.

[8] FUKASAWA, R.; LYSGAARD, J.; ARAGÃO, M. P.; REIS, M.; UCHOA, E.; WERNECK, R. F. Robust Branch-and-Cut-and-Price for the Capacitated Vehicle Routing Problem. **Mathematical Programming**, v. 106, n. 3, p. 491-511, 2006.

[9] GILLETT, B.; JOHNSON, J. Multi-terminal vehicle-dispatch algorithm. **Omega**, v. 4, p. 711–717, 1976.

[10] HAKIMI, S. L. Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems. **Operations Research**, v. 13, p. 462-475, 1965.

[11] JUAN, A.; FAULÍN, J.; CABALLÉ, S., BARRIOS, B.; RUIZ, R. The SR-GCWS hybrid alogorithm for solving the capacitated vehicle routing problem. **Applied Soft Computing Journal**. no 1, p. 215-224. 2010.

[12] LAPORTE, G.; GENDREAU, M.; POTVIN, J. Y.; SEMET, F. Classical and modern heuristics for the vehicle routing problem. **International Transactions in Operational Research**, v. 7, n. 4-5, p. 285-300, 1999.

[13] LEIJEN, D.; HALL, J. Otimize o código gerenciado para máquinas de vários núcleos. **MSDN Magazine**, p. 31-33, Outubro 2007.

[14] LIN, S.; KERNIGHAN, B. W. An Effective Heuristic Algorithm for the Traveling Salesman Problem, **Operations Research**, v. 21, p. 498-516, 1973.

[15] LIN, S. W.; LEE, Z. J.; YING, K. C.; LEE, C. Y. Applying hybrid meta-heuristics for capacitated vehicle routing problem. **Expert Systems with Applications**, v. 36, p. 1505–1512, 2009.

[16] SARIKLIS D.; POWELL S. A heuristic method for the open vehicle routing problem. **Journal of the Operational Research Society**, v. 51, p.564–573, 2000.

[17] STEINER, M. T. A.; ZAMBONI, L. V. S.; COSTA, D. M. B.; CARNIERI, C.; SILVA, A. C. L. O problema de roteamento no transporte escolar. **Pesquisa Operacional**, v. 20, p. 83-99, 2000.

[18] TOUB, S. O passado, o presente e o futuro da paralelização de aplicativos .NET. **MSDN Magazine**, p.48-52, Agosto 2011