MÉTODO DE PESQUISA EM VIZINHANÇA VARIÁVEL APLICADO À RESOLUÇÃO DO PROBLEMA DE ROTEAMENTO DE VEÍCULOS

Dárlinton Barbosa Feres Carvalho
Departamento de Computação
Universidade Federal de Ouro Preto
35.400-000 Ouro Preto, MG
dioncons@iceb.ufop.br

Guilherme Arantes de Oliveira
Departamento de Computação
Universidade Federal de Ouro Preto
35.400-000 Ouro Preto, MG
guilhermeaol@yahoo.com.br

Marcone Jamilson Freitas Souza
Departamento de Computação
Universidade Federal de Ouro Preto
35.400-000 Ouro Preto, MG
marcone@iceb.ufop.br

Resumo

Neste trabalho apresenta-se um método heurístico, baseado em GRASP e Método de Pesquisa em Vizinhança Variável (VNS), para resolver o Problema de Roteamento de Veículos. Uma solução inicial é gerada pela fase de construção GRASP seguida de um refinamento pelo método VNS, o qual, por sua vez, utiliza o Método de Descida em Vizinhança Variável como método de busca local. Resultados computacionais são apresentados para um conjunto de problemas-teste encontrados na literatura. O método proposto é de fácil entendimento e implementação, requer a manipulação de poucos parâmetros, produz soluções de boa qualidade rapidamente e é capaz de melhorar essas soluções quando lhe é dado um tempo de processamento mais elevado.
Palavras-chave: Roteamento de Veículos, Método de Pesquisa em Vizinhança Variável, Metaheurísticas.

Abstract

In this work we present a heuristic approach, based on GRASP and Variable Neighborhood Search (VNS), for solving the Vehicle Routing Problem (VRP). Initially a solution is generated by the GRASP construction phase. This solution is refined by the VNS method, which uses the Variable Neighborhood Descent Method for local search. The method was tested using several benchmarks of VRP found in literature. The proposed method is easy to understand and to code, needs few parameters, produces good quality solutions quickly and it is able to improve them when there is a higher processing time.
Keywords: Vehicle Routing, Variable Neighborhood Search, Metaheuristics.

1 Introdução

O Problema de Roteamento de Veículos (PRV) pode ser definido como segue. Dado um conjunto de cidades (ou consumidores), cada qual com uma demanda q_i por um produto, e um depósito com veículos de capacidade Q, encontrar as rotas para os veículos minimizando os custos de transporte.

Uma grande quantidade de aplicações práticas do PRV pode ser encontrada na literatura. Por exemplo, Brown & Graves (1981), Fisher et al. (1982), Bell et al. (1983), Evans & Norback (1985), Golden & Watts (1987) mostram aplicações nas indústrias de petróleo, químicas, alimentícias e de bebidas.

O interesse no PRV é parcialmente devido à sua importância prática, mas também à sua dificuldade. Como uma generalização do Problema do Caixeiro Viajante (PCV), o PRV pertence à classe de problemas NP-Difícil (LENSTRA, 1981), portanto não existem algoritmos em tempo polinomial para encontrar soluções ótimas. Os algoritmos exatos existentes raramente conseguem resolver problemas envolvendo mais do que 50 consumidores (RENAUD & BOCTOR, 2002).

Devido ao limitado sucesso dos métodos exatos, os esforços de pesquisa têm sido direcionados no desenvolvimento de heurísticas para lidar com problemas de maior porte. Exemplos de heurísticas bem sucedidas para resolver PRV são os algoritmos baseados em Busca Tabu de Taillard (1993), Osman (1993) e Gendreau et al. (1994), e a heurística de pétalas de Renaud et al.(1996).

Para uma revisão das mais importantes heurísticas clássicas e modernas para o PRV veja Cordeau et al. (2002) e Laporte (1992). Uma bibliografia do PRV pode ser obtida em Laporte & Osman (1995).

Neste artigo propomos um método de duas fases para a resolução do PRV, que combina a fase de construção GRASP, com um refinamento usando o Método de Pesquisa em Vizinhança Variável (VNS). Para obter uma solução inicial, como será mostrado na seção 6, o número de veículos inicialmente considerado é aquele necessário para construir rotas viáveis de cada veículo. A partir do refinamento da solução, baseado em função de avaliação que procura minimizar as distâncias percorridas, esse número de veículos pode ser reduzido. As estruturas de vizinhança utilizadas no VNS são simples e proporcionam alterações na solução capazes de escapar de ótimos locais, conforme demonstrado nos testes. O método para busca local no VNS é o Método de Descida em Vizinhança Variável (VND) que também foi projetado com estruturas simples e de baixa complexidade para uma rápida execução.

O método proposto foi testado usando-se um conjunto de instâncias clássicas encontradas na literatura. Resultados computacionais demonstram a eficiência do método na obtenção de soluções finais de qualidade próxima aos melhores valores encontrados na literatura.

Este trabalho está organizado como segue. Na seção 2 formula-se o PRV. Na seção 3 mostra-se como uma solução do PRV é representada. As diferentes estruturas de vizinhança consideradas e a função de avaliação de uma solução são apresentadas nas seções 4 e 5, respectivamente. Na seção 6 mostra-se como uma solução inicial para o PRV é construída. As seções 7 e 8 descrevem as adaptações dos métodos de Pesquisa em Vizinhança Variável e Descida em Vizinhança Variável aplicados ao PRV. Na seção 9 apresenta-se o método heurístico GRASP_VNS proposto. A seção 10 apresenta um modelo de programação matemática para otimizar as rotas de cada veículo em uma solução do PRV. Na seção 11 apresentam-se os resultados obtidos pela aplicação do método proposto a um conjunto de instâncias clássicas do problema. A seção 12 conclui o trabalho.

2 Formulação do Problema do Roteamento de Veículos (PRV)

Seja $G = (V, E)$ um grafo não direcionado, onde $V = \{v_0, v_1,..., v_n\}$ é o conjunto dos vértices e $E = \{(v_i, v_j): v_i, v_j \in V, i < j\}$ é o conjunto de arestas. O vértice v_0 representa o depósito, sendo este a base de uma frota de veículos idênticos de capacidade Q, enquanto os vértices remanescentes correspondem às cidades ou consumidores. Cada consumidor v_i tem uma

demanda não negativa q_i e $q_0 = 0$. Neste trabalho supõe-se que existe um número ilimitado de veículos no depósito.

A cada aresta (v_i, v_j) está associada uma distância não negativa c_{ij} que representa a distância entre os consumidores.

O Problema de Roteamento de Veículos consiste em determinar o conjunto de rotas que deverão ser feitas pelos veículos minimizando os custos de transporte, dado pela distância e respeitando as seguintes condições:

(a) Cada rota começa e termina no depósito;
(b) Toda cidade de $V\backslash\{v_0\}$ é visitada somente uma vez por somente um veículo;
(c) A demanda total de qualquer rota não deve superar a capacidade Q de um veículo.

3 Representação do PRV

Assumimos a representação usada por Pradenas & Parada (1999). Uma solução do PRV é representada por meio de uma permutação de cidades, numeradas de 1 a n, separadas em tantas partições quantos forem o número de veículos usados. O elemento separador é representado pelo valor zero e indica o depósito.

Por exemplo, se há 6 consumidores, 3 veículos e a solução s é {0-3-4-0-1-5-2-0-6-0} então as rotas dos veículos, denominadas pétalas, são {0-3-4-0}, {0-1-5-2-0} e {0-6-0}.

4 Estruturas de vizinhança

Seja S o conjunto das soluções para o PRV. A fim de projetar um algoritmo baseado na exploração de vizinhanças variáveis para resolver o PRV, faz-se necessário definir diversas estruturas de vizinhança, isto é, funções N que associam um conjunto de soluções $N(s)$ com cada solução $s \in S$ obtida por uma modificação parcial de s, chamada movimento. Consideramos seis estruturas de vizinhança, a saber: N^1, N^2, N^3, N^4, N^5, N^6.

O primeiro movimento consiste na troca de dois números inteiros em uma mesma pétala da solução. Estes números representam apenas os consumidores. O segundo, terceiro, quarto, quinto e sexto movimentos consistem em efetuar uma, duas, três, quatro ou cinco trocas, respectivamente, entre quaisquer elementos da solução. Portanto, no primeiro movimento são realizadas somente trocas entre consumidores pertencentes a uma mesma pétala, enquanto que nos demais movimentos permite-se a troca entre consumidores pertencentes a pétalas diferentes, bem como uma troca entre um consumidor e o depósito, resultando na alteração do retorno ao depósito.

A vizinhança $N^1(s)$ de uma dada solução s é o conjunto de todos os vizinhos s' gerados pelo primeiro movimento. Por exemplo, $s' = \{0\text{-}3\text{-}4\text{-}0\text{-}1\text{-}\mathbf{2}\text{-}\mathbf{5}\text{-}0\text{-}6\text{-}0\} \in N^1(s)$.

A vizinhança $N^2(s)$ de uma dada solução s é o conjunto de todos os vizinhos s' gerados pelo segundo movimento. Por exemplo, $s' = \{0\text{-}3\text{-}\mathbf{5}\text{-}0\text{-}1\text{-}\mathbf{4}\text{-}2\text{-}0\text{-}6\text{-}0\} \in N^2(s)$.

A vizinhança $N^3(s)$ de uma dada solução s é o conjunto de todos os vizinhos s' gerados pelo terceiro movimento. Por exemplo, $s' = \{0\text{-}3\text{-}\mathbf{5}\text{-}0\text{-}\mathbf{6}\text{-}\mathbf{4}\text{-}2\text{-}0\text{-}\mathbf{1}\text{-}0\} \in N^3(s)$.

A vizinhança $N^4(s)$ de uma dada solução s é o conjunto de todos os vizinhos s' gerados pelo quarto movimento. Por exemplo, $s' = \{0\text{-}3\text{-}\mathbf{5}\text{-}\mathbf{6}\text{-}0\text{-}\mathbf{4}\text{-}2\text{-}0\text{-}\mathbf{1}\text{-}0\} \in N^4(s)$.

A vizinhança $N^5(s)$ de uma dada solução s é o conjunto de todos os vizinhos s' gerados pelo quinto movimento. Por exemplo, $s' = \{0\text{-}3\text{-}\mathbf{5}\text{-}\mathbf{6}\text{-}0\text{-}\mathbf{4}\text{-}\mathbf{0}\text{-}\mathbf{2}\text{-}\mathbf{1}\text{-}0\} \in N^5(s)$.

A vizinhança $N^6(s)$ de uma dada solução s é o conjunto de todos os vizinhos s' gerados pelo sexto movimento. Por exemplo, $s' = \{0\text{-}\mathbf{4}\text{-}\mathbf{5}\text{-}\mathbf{6}\text{-}0\text{-}\mathbf{3}\text{-}\mathbf{0}\text{-}\mathbf{2}\text{-}\mathbf{1}\text{-}0\} \in N^6(s)$.

5 Função Objetivo

A fim de avaliar a solução do PRV usamos uma função objetivo baseada em penalização. Mais especificamente, seja $f_1(s)$ representando a função objetivo pura da solução s, isto é, a soma das distâncias percorridas por todos os veículos, e $O(s)$ o total das sobrecargas dos veículos associada a esta solução, caso exista. O algoritmo desenvolvido trabalha com a função objetivo $f(s) = f_1(s) + \beta \times O(s)$, onde β é um fator de penalidade não negativo.

6 Construção de uma solução inicial

Para construir uma solução inicial utilizamos a fase de construção do método GRASP (Procedimento de busca adaptativa gulosa e randomizada). Este é um método iterativo proposto por Feo & Resende (1995), que consiste de duas fases: uma fase de construção, na qual uma solução é gerada elemento a elemento e de uma fase de busca local, na qual um ótimo local na vizinhança da solução construída é pesquisado. Essas duas fases são aplicadas durante um certo número de iterações. A melhor das soluções obtidas ao final dessas iterações é retornada como solução final. No método proposto, como será mostrado na seção 9, essas duas fases são aplicadas uma única vez, sendo a fase de refinamento realizada pelo método VNS. Mostramos, a seguir, como construir uma solução inicial para o PRV.

Inicialmente, o depósito é adicionado à solução, pois é o ponto de partida dos veículos. A cada iteração da fase de construção, os próximos elementos candidatos a serem incluídos na solução, ou seja, as cidades (consumidores), são colocados em uma lista de candidatos (LC), seguindo um critério de ordenação relativo à distância de cada um ao último elemento adicionado à solução. Esse processo de seleção é uma heurística adaptativa gulosa, que estima o benefício da seleção de cada um dos elementos. A heurística é adaptativa porque os benefícios associados com a escolha de cada elemento são atualizados em cada iteração da fase de construção para refletir as mudanças oriundas da seleção do elemento anterior. A componente probabilística do procedimento reside no fato de que cada elemento é selecionado de forma aleatória a partir de um subconjunto restrito formado pelos melhores elementos da lista de candidatos, chamada de lista de candidatos restrita (LCR). O tamanho da LCR é definido segundo um fator $\alpha \in [0,1]$, tal que $|LCR| = \alpha \times |LC|$. A cada consumidor selecionado é verificada a viabilidade de sua inclusão na solução. Caso não seja viável incluí-lo na solução _ o que acontece quando a capacidade do veículo é superada _ insere-se um zero, relativo ao depósito, significando o fim de uma rota e início de outra. Essa técnica de escolha permite que diferentes soluções sejam geradas em cada execução deste procedimento de construção GRASP.

No caso de se utilizar um número mínimo de veículos, o procedimento acima pode ser facilmente modificado. Basta para tanto proceder do mesmo modo descrito anteriormente, mas permitir a inviabilidade na rota do último veículo.

7 Método de Pesquisa em Vizinhança Variável

O Método de Pesquisa em Vizinhança Variável, conhecido como VNS (do termo em inglês *Variable Neighborhood Search*) é um método de busca local proposto por Mladenovic & Hansen (1997) que consiste em explorar o espaço de soluções através de trocas sistemáticas de estruturas de vizinhança. Diferentemente de outras metaheurísticas baseadas em métodos de busca local, o método VNS não segue uma trajetória, mas explora vizinhanças gradativamente mais "distantes" da solução corrente e focaliza a busca em torno de uma nova solução somente se um movimento de melhora é realizado. O método inclui, também, uma rotina de busca local que pode usar diferentes estruturas de vizinhança.

A Figura 1 mostra o pseudo-código do método VNS aplicado à resolução do PRV. O método proposto tem como critério de parada um tempo máximo de processamento igual a t, utiliza as seis estruturas de vizinhança definidas na seção 4 e tem o Método de Descida em Vizinhança Variável (vide seção 8) como método de busca local. É importante observar que no método proposto são utilizadas 6 estruturas diferentes de vizinhança, sendo algumas das quais repetidas. A estrutura N^5 é repetida 2 vezes e a estrutura N^6 repetida 4 vezes. Este arranjo repetindo vizinhanças é proposto para privilegiar os movimentos que provocam maior variabilidade na solução. Com isto tende-se a aumentar a probabilidade de se escapar de ótimos locais, situação que pode ocorrer a partir da escolha aleatória de um vizinho da melhor solução corrente. Nesta figura vizinho_qualquer(s, N) é uma função que retorna uma solução aleatória vizinha de s na estrutura $N^{(k)}$.

```
procedimento VNS(s, t);
1       enquanto ( tempo_execução < t )
2               k ← 1;
3               enquanto ( k <= 10 )
4                       escolha(k)
5                               1: s'←vizinho_qualquer(s, N^1);
6                               2: s'←vizinho_qualquer(s, N^2);
7                               3: s'←vizinho_qualquer(s, N^3);
8                               4: s'←vizinho_qualquer(s, N^4);
9                               5: s'←vizinho_qualquer(s, N^5);
10                              6: s'←vizinho_qualquer(s, N^5);
11                              7: s'←vizinho_qualquer(s, N^6);
12                              8: s'←vizinho_qualquer(s, N^6);
13                              9: s'←vizinho_qualquer(s, N^6);
14                             10: s'←vizinho_qualquer(s, N^6);
15                      fim escolha;
16                      s'←VND(s');
17                      se ( f(s') < f(s))
18                              s ← s';
19                              k ← 1;
20                      senão
21                              k ← k + 1;
22                      fim-se;
23              fim enquanto;
24      fim enquanto;
25      Retorne s;      {Retorne a melhor solução}
fim VNS;
```

Figura 1: Método VNS aplicado ao PRV

8 Método de Descida em Vizinhança Variável

O Método de Descida em Vizinhança Variável, conhecido como VND (do termo em inglês *Variable Neighborhood Descent*) é um método de busca local proposto por Mladenovic & Hansen (1997) que consiste em explorar o espaço de soluções através de trocas sistemáticas de estruturas de vizinhança, aceitando somente soluções de melhora da solução corrente e retornando à primeira estrutura quando uma solução melhor é encontrada.

No VND aplicado à resolução do PRV foram consideradas apenas as estruturas de vizinhança N^1 e N^2. As pesquisas nas estruturas de vizinhança N^3 a N^6 não foram consideradas por terem alta complexidade e implicarem na degradação da performance do método.

A Figura 2 mostra o pseudo-código do método VND. Nesta figura, descida_1opt é um método de descida que utiliza a estrutura de vizinhança N^1; enquanto descida_2opt é um método de descida que faz uso da estrutura de vizinhança N^2.

```
procedimento VND(s);
1       s'← s;
2       k ← 1;
3       enquanto ( k <= 2 )
4               escolha(k)
5                       1: s'←descida_1opt(s');
6                       2: s'← descida_2opt (s');
7               fim escolha;
8               se ( f(s') < f(s))
9                       s ← s';
10                      k ← 1;
11              senão
12                      k ← k + 1;
13              fim-se;
14      fim enquanto;
15      Retorne s;      {Retorne a melhor solução}
fim VND;
```

Figura 2: Método VND aplicado ao PRV

9 Método proposto

O método proposto é um algoritmo de duas fases. Na primeira, uma solução é gerada pela fase de construção GRASP com um fator α de aleatoriedade (vide seção 6). Na segunda fase esta solução é refinada pelo método VNS em um tempo de execução t (vide seção 7).

A Figura 3 ilustra o pseudo-código do método proposto.

```
procedimento GRASP_VNS(α, t);
1       s ← ConstruaSolucaoInicial(α);
2       s ← VNS(s, t);
3       Retorne s;      {Retorne a melhor solução}
fim GRASP_VNS;
```

Figura 3: Método proposto

10 Método Exato para otimizar cada pétala

Para encontrar a melhor seqüência de visita para cada veículo é necessário resolver o Problema do Caixeiro Viajante (PCV). Seja V_r um subconjunto de V incluindo o depósito, isto é, uma permutação de um conjunto de cidades iniciando do depósito (uma pétala da solução do PRV). Uma formulação de programação matemática citada em Laporte (1992) é apresentada abaixo:

$$\min \sum_{i \in V_r} \sum_{\substack{j \in V_r \\ j \neq i}} c_{ij} x_{ij}$$

$$\sum_{\substack{i \in V_r \\ i \neq j}} x_{ij} = 1 \quad \forall j \in V_r$$

$$\sum_{\substack{j \in V_r \\ j \neq i}} x_{ij} = 1 \quad \forall i \in V_r$$

$$\sum_{\substack{i \in V_r \\ i \neq j}} y_{ji} - \sum_{\substack{i \in V_r \\ i \neq j}} y_{ij} = 1 \quad \forall j \in V_r \setminus \{v_0\}$$

$$y_{ij} \leq (n-1) x_{ij} \quad \forall i, j \in V$$

$$x_{ij} \in \{0,1\} \quad \forall i, j \in V_r$$

$$y_{ij} \geq 0 \quad \forall i, j \in V_r$$

11 Experimentos Computacionais

O método proposto foi codificado na linguagem C usando o compilador Borland C++ Builder 6.0 e testado em 8 instâncias clássicas do PRV encontradas na literatura no web site http://ina.eivd.ch/collaborateurs/etd/problemes.dir/vrp.dir/vrp.html.

Todos os experimentos foram realizados em um PC com processador Pentium IV, de 1.8 GHz, 512 MB de RAM rodando a plataforma Windows XP.

Para cada teste foram realizadas 25 execuções, cada qual partindo de uma semente diferente de números aleatórios. As soluções finais das instâncias c50, c75, c100 e c100b foram também submetidas ao método exato descrito na seção 10 para otimizar as rotas de cada veículo. O otimizador utilizado foi o LINGO, versão 7, através da chamada de sua DLL no programa. Quando a aplicação do método exato resultou na melhora da solução final produzida pelo método GRASP_VNS proposto, foram relatados os dados relativos à solução obtida com e sem a aplicação do método exato, sendo acrescido ao tempo destinado à execução do método heurístico a média do tempo de processamento requerido pelo método exato.

A Tabela 1 mostra os resultados encontrados para diferentes tempos de processamento e fatores de aleatoriedade na construção de uma solução inicial. Nesta tabela, "#cid" representa o número de cidades da instância, "Cap. vei." indica a capacidade dos veículos, "α" o valor do fator de aleatoriedade da fase de construção GRASP, "ME" sinaliza se a aplicação do método exato resultou na melhora da solução final do método GRASP_VNS, "#veic*" é o número de veículos utilizado na melhor solução encontrada, "Média #veic" é a média do número de veículos utilizados nas soluções e "desvio" significa a percentagem de desvio entre o valor médio ($\bar{f}$) encontrado pelo método em 25 execuções e o melhor valor conhecido na literatura para a instância considerada (f^*), isto é:

$$\text{Desvio} = \frac{\left(\bar{f} - f^*\right) \times 100}{f^*}$$

Teste	Instância	# cid	Cap. vei.	Melhor Valor conhecido	Tempo CPU (seg)	α	ME	GRASP_VNS				
								Melhor Valor	Valor Médio	Desvio (%)	# veic*	Média # veic
1	c50.dat	50	160	524.61	150	0,01	-	524.61	532.87	1.57	5	5
2	c50.dat	50	160	524.61	150	0,35	-	527.67	545.99	4.07	5	5.44
3	c50.dat	50	160	524.61	300	0,35	-	524.61	541.24	3.17	5	5.28
4	c75.dat	75	140	835.26	300	0,01	-	842.96	865.56	3.62	10	10.96
5	c75.dat	75	140	835.26	150	0,35	-	844.44	875.54	4.82	11	10.92
6	c75.dat	75	140	835.26	156	0,35	✓	844.44	875.20	4.78	11	10.92
7	c75.dat	75	140	835.26	300	0,35	-	840.40	871.47	4.33	10	10.88
8	c75.dat	75	140	835.26	600	0,35	-	846.59	866.99	3.80	11	10.92
9	c75.dat	75	140	835.26	900	0,35	-	845.25	865.24	3.59	10	10.92
10	c100.dat	100	200	826.14	300	0,01	-	853.01	879,44	6.45	8	8
11	c100.dat	100	200	826.14	334	0,01	✓	851.72	875.19	5.93	8	8
12	c100.dat	100	200	826.14	300	0,35	-	862.23	887.14	7.38	8	8
13	c100.dat	100	200	826.14	340	0,35	✓	860.55	885.66	7.20	8	8
14	c100.dat	100	200	826.14	1500	0,35	-	842.98	868.83	5.16	8	8
15	c100.dat	100	200	826.14	1534	0,35	✓	842.98	868.30	5.10	8	8
16	c100b.dat	100	200	819.56	300	0,01	-	828,97	831,09	1,40	10	10
17	c100b.dat	100	200	819.56	318	0,01	✓	828.93	828.93	1.14	10	10
18	c100b.dat	100	200	819.56	300	0,35	-	850.45	935.50	14.14	10	10
19	c100b.dat	100	200	819.56	944	0,35	✓	847.31	933.98	13.96	10	10
20	c100b.dat	100	200	819.56	1500	0,35	-	820.48	893.10	8.97	10	10
21	c100b.dat	100	200	819.56	1742	0,35	✓	820.48	890.94	8.70	10	10
22	tai75a.dat	75	1445	1618.36	150	0,01	-	1624.16	1656.58	2.36	10	10.36
23	tai75a.dat	75	1445	1618.36	150	0,10	-	1622.61	1652.21	2.09	10	10.32
24	tai75a.dat	75	1445	1618.36	150	0,35	-	1628.69	1665.83	2.93	10	10.4
25	tai75a.dat	75	1445	1618.36	300	0,35	-	1619.52	1652.43	2.10	10	10.4
26	tai75a.dat	75	1445	1618.36	600	0,35	-	1622.50	1643.33	1.54	10	10.2
27	tai75b.dat	75	1679	1344.64	150	0,01	-	1349.86	1362.39	1.31	10	10
28	tai75b.dat	75	1679	1344.64	150	0,10	-	1349.52	1367.36	1.69	10	10
29	tai75b.dat	75	1679	1344.64	150	0,35	-	1349.72	1374.29	2.20	10	10
30	tai75b.dat	75	1679	1344.64	300	0,35	-	1345.62	1365.31	1.53	10	10
31	tai75b.dat	75	1679	1344.64	600	0,35	-	1346.63	1360.77	1.20	10	10
32	tai75c.dat	75	1122	1291.01	150	0,01	-	1334.11	1375.10	6.51	9	9.76
33	tai75c.dat	75	1122	1291.01	150	0,10	-	1302.21	1353.36	4.82	9	9.76
34	tai75c.dat	75	1122	1291.01	150	0,35	-	1309.27	1390.38	7.69	9	9.4
35	tai75c.dat	75	1122	1291.01	300	0,35	-	1314.91	1370.26	6.13	9	9
36	tai75c.dat	75	1122	1291.01	600	0,35	-	1295.1	1352.26	4.74	9	9.12
37	tai75d.dat	75	1699	1365.42	150	0,01	-	1388.23	1406.55	3.01	9	9.68
38	tai75d.dat	75	1699	1365.42	150	0,10	-	1372.26	1406.00	2.97	9	9.52
39	tai75d.dat	75	1699	1365.42	150	0,35	-	1377.41	1420.07	4.00	9	9.32
40	tai75d.dat	75	1699	1365.42	300	0,35	-	1379.76	1410.62	3.31	9	9.36
41	tai75d.dat	75	1699	1365.42	600	0,35	-	1378.25	1405.91	2.96	9	9.28

Tabela 1: Resultados Computacionais

12 Conclusões

Este artigo apresenta um método simples baseado nas metaheurísticas GRASP e VNS para resolver o Problema de Roteamento de Veículos. A idéia básica é a geração de diferentes estruturas de vizinhança que devem ser usadas em uma ordem progressiva de complexidade. O método proposto requer poucos parâmetros, basicamente o fator de aleatoriedade relativo à construção de uma solução inicial, o tempo de processamento e a seqüência de estruturas de vizinhança. A partir desses parâmetros pode-se representar características do problema que irão guiar o método e muitas vezes determinar a qualidade da solução. Por exemplo, partindo de um conhecimento prévio do problema onde os consumidores estão agrupados em regiões distantes entre si (*clustered*), como no problema teste c100b.dat, é conveniente adotar um critério de baixa aleatoriedade na construção da solução inicial. Em outros, tais como em tai75a.dat, um índice de aleatoriedade um pouco mais elevado resulta em soluções finais de melhor qualidade na média. A utilização do método exato para solução do PCV nas sub-rotas garante um aumento da qualidade da solução gerada em problemas com baixo tempo de processamento ou com número elevado de consumidores.

Este trabalha contribui com a apresentação de um método de fácil entendimento e implementação, que requer a manipulação de poucos parâmetros e produz soluções de boa qualidade rapidamente, sendo capaz de melhorar essas soluções quando lhe é dado um tempo de processamento mais elevado.

References

BELL, W.; DALBERTO, L.; FISHER, M. L.; GREENFIELD, A.; JAIKUMAR, R.; MACK, R. & PRUTZMAN, P. (1983), Improving distribution of industrial gases with an on-line computerized routing and scheduling systems, *Interfaces*, v. 13, p. 4-23.

BROWN, G. & GRAVES, G. (1981), Real-time dispatch of petroleum tank trunks, *Management Science,* v. 27, p. 19-32.

CORDEAU, J. F.; GENDREAU, M.; LAPORTE, G.; POTVIN, J. Y. & SEMET, F. (2002), A guide to vehicle routing heuristics, *Journal of the Operational Research Society*, v. 53, p. 512-522.

EVANS, S & NORBACK, J (1985), The impact of a decision-support system for vehicle routing in a food service supply situation, *Journal of the Operational Research Society*, v. 36, p. 467-472.

FEO, T. A. & RESENDE, M. G. C. (1995), Greedy randomized adaptive search procedures, *Journal of Global Optimization*, v. 6, p. 109-133.

FISHER, M. L.; GREENFIELD, R.; JAIKUMAR, R. & LESTER, J. (1982), A computerized vehicle routing application, *Interfaces*, v. 1, p. 45-52.

FISHER, M. L. & JAIKUMAR, R. (1981), A generalized assignment heuristic for vehicle routing, *Networks*, v. 11, p. 109-124.

GENDREAU, M.; HERTZ, A. & LAPORTE, G. (1994), A tabu search heuristic for the vehicle routing problem, *Management Science*, v. 40, p. 1276-1290.

GOLDEN, B. L. & WATTS, E. (1987), Computerized vehicle routing in the soft drink industry, *Operations Research*, v. 35, p. 6-17.

LAPORTE, G. (1992), The Vehicle-Routing Problem – An Overview of exact and approximate algorithms, *European Journal of Operational Research*, v. 59, p. 345-358.

LAPORTE, G & OSMAN, I. H. (1995), Routing Problems: A bibliography, *Annals of Operations Research*, v. 61, p. 227-262.

LENSTRA, J. & RINNOOY KAN, A. (1981), Complexity of vehicle routing and scheduling problems, *Networks*, v.11, p. 221-228.

MLADENOVIC, N. & HANSEN, P. (1997) - Variable neighborhood Search. *Computers and Operations Research*, v. 24, p. 1097 – 1100.

PRADENAS, L. & PARADA, V. (1999), Use of Genetic Algorithms, Tabu Search and Simulated Annealing in the Vehicle Routing Problem, *Proceedings of the Third Metaheuristics International Conference (MIC'99),* p. 371-374.

OSMAN, I. H. (1993), Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem, *Annals of Operations Research 41*, 421-451.

RENAUD, J.; BOCTOR, F. F. &. LAPORTE, G. (1996), An improved petal heuristic for the vehicle routing problem, *Journal of the Operational Research Society,* v. 47, p. 329-336.

RENAUD, J. & BOCTOR, F. F. (2002), A sweep-based algorithm for the fleet size and mix vehicle routing problem, *European Journal of Operational Research,* v. 140, p. 618-628.

TAILLARD, E. D. (1993), Parallel iterative search methods for vehicle routing problems, *Networks 23*, 661-673.